

(11)特許出願公表番号
特表2002-518899
(P2002-518899A)

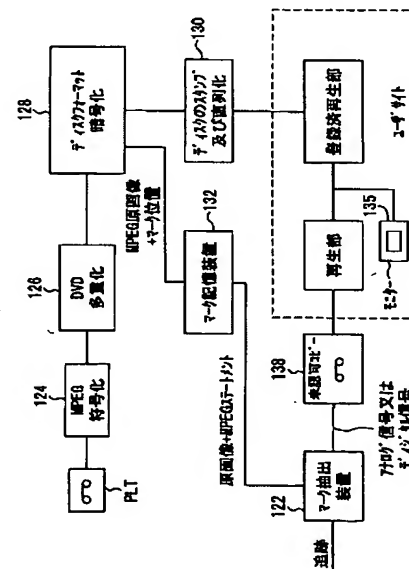
(43)公表日 平成14年6月25日(2002.6.25)

(51)Int.Cl. ⁷	識別記号	F I	テームコード [*] (参考)
H 0 4 N 5/91		G 1 1 B 20/10	H 5 C 0 5 3
G 1 1 B 20/10		20/12	5 C 0 5 9
20/12		H 0 4 N 5/91	P 5 D 0 4 4
H 0 4 N 5/92		5/92	H
7/24		7/13	Z
		審査請求 未請求 予備審査請求 有 (全110頁)	
(21)出願番号	特願2000-554137(P2000-554137)	(71)出願人	ディジタルビデオエクスプレス エル. ビー.
(86) (22)出願日	平成11年6月7日(1999.6.7)		アメリカ合衆国 バージニア州 20170、
(85)翻訳文提出日	平成12年12月8日(2000.12.8)		ハーンドン、ハーンドンパークウェイ
(86)国際出願番号	P C T / U S 9 9 / 1 1 7 9 7		580
(87)国際公開番号	W O 9 9 / 6 5 2 4 1	(72)発明者	ロバート ダブリュー. シューマン
(87)国際公開日	平成11年12月16日(1999.12.16)		アメリカ合衆国 バージニア州 22124
(31)優先権主張番号	0 9 / 0 9 2 , 8 9 8		オークトン、クエイロード 11723
(32)優先日	平成10年6月8日(1998.6.8)	(72)発明者	シュレオン イウ
(33)優先権主張国	米国 (U S)		アメリカ合衆国 バージニア州 22043
			フォールズチャーチ、202、ピンミットラ
			ンレーン 2252
		(74)代理人	弁理士 前田 弘 (外7名)
			最終頁に続く

(54) 【発明の名称】 動画像等の前もって記録された所有権主張材料の非認可コピーの出所を追跡する装置及び方法

(57) 【要約】

ビデオDVD等の情報含有媒体の非認可コピーの出所を追跡可能にするため、媒体に記録された圧縮デジタル信号ストリームを、フレーム毎に位置が変化しコピーが製造された場所と時間を示す情報で符合化された画素ブロックを有する「走行マーク」を含むように変更する。検査対象媒体を元のビデオ信号と走行マークの位置とを含む基準媒体と同時に再生する。コピーメッセージの出所を有利に暗号化及びスクランブル化してコピー者による検出と変更を防止する。メッセージを、例えば、スペクトラム拡散キャリアに拡散させて、多数の出所の中からの識別性を向上させるとともに、ノイズが存在する中での復調を可能にする。



BEST AVAILABLE COPY

【特許請求の範囲】

【請求項1】 情報信号源により生成された情報含有圧縮デジタル信号ストリームを処理して、オリジナル記録媒体に記録し、情報含有信号ストリームの情報内容に印を付けてオリジナル記録媒体からの情報含有圧縮デジタル信号ストリームの非認可コピーの出所を追跡可能にする方法であって、

上記オリジナル記録媒体の再生時に元の圧縮デジタル信号ストリームを受け取る工程と、

上記オリジナル記録媒体から上記元の圧縮デジタル信号ストリームの再生の出所に対応するコピー源メッセージを得る工程と、

上記コピー源メッセージを含むよう上記元の圧縮デジタル信号ストリームを修正することにより修正信号ストリームを生成する工程とを備えた方法。

【請求項2】 上記修正信号ストリームが第2の記録媒体で再生される請求項1記載の方法であって、

上記第2の記録媒体の再生時に上記修正信号ストリームを再生する工程と、

上記元の情報含有圧縮デジタル信号ストリームを上記修正信号ストリームと比較して上記コピー源メッセージを抽出する工程とを備えた方法。

【請求項3】 上記情報含有圧縮デジタル信号ストリームはビデオ信号であり、上記圧縮デジタル信号ストリーム修正工程は上記圧縮デジタル信号ストリームにより生成された画像の特徴を終始するものである請求項1記載の方法。

【請求項4】 上記圧縮デジタル信号はMPEG圧縮標準で前もって符号化される請求項3記載の方法。

【請求項5】 修正用にビデオフレームから少なくとも一つの領域を選択し、その選択されたビデオフレーム領域を修正する工程を備えている請求項4記載の方法。

【請求項6】 選択されたビデオフレーム領域のアドレスを格納する工程を備えている請求項5記載の方法。

【請求項7】 上記選択はビデオフレーム内の画像の画像属性に基づいて実行される請求項6記載の方法。

【請求項 8】 上記画像属性は画像境界線である請求項 7 記載の方法。

【請求項 9】 上記選択されたフレーム領域を上記コピー源メッセージに対応する修正フレーム領域に置換する工程を備えている請求項 6 記載の方法。

【請求項 10】 上記選択されたフレーム領域を修正フレーム領域への置換の前に調整する工程を備えている請求項 9 記載の方法。

【請求項 11】 上記コピー源メッセージに応じて連続するフレームから修正用に様々な領域を選択する工程を備えている請求項 5 記載の方法。

【請求項 12】 上記フレーム領域は画素ブロックまたはマクロブロックを備えている請求項 10 記載の方法。

【請求項 13】 上記圧縮デジタル信号ストリームの修正前に上記コピー源メッセージを暗号化する工程を備えている請求項 1 記載の方法。

【請求項 14】 上記圧縮デジタル信号ストリームの修正前に上記コピー源メッセージをスクランブル化する工程を備えている請求項 1 記載の方法。

【請求項 15】 上記圧縮デジタル信号ストリームの修正前に上記コピー源メッセージを符号化してスペクトラム拡散キャリアを生成する工程を備えている請求項 1 記載の方法。

【請求項 16】 予め記録されたオリジナル媒体から再生されたオリジナル M P E G 符号化ビデオ信号ストリーム及び／または音声信号ストリームの形で圧縮デジタル情報を処理して上記媒体からの信号ストリームの非認可コピーを追跡可能にする方法であって、

M P E G 標準形式の情報とその後の再生を記述する情報に関するデータビット用のビデオフレーム内の前選択された画素ブロック位置を格納するオリジナル媒体を生成する工程と、

上記オリジナル媒体のその後の再生時に M P E G データストリームを受け取る工程と、

上記 M P E G 信号ストリームを修正することにより修正信号ストリームを生成して上記その後の再生情報を上記前選択された画素ブロック位置に対応する画素ブロックに埋め込む工程とを備えている方法。

【請求項 17】 上記元の信号ストリームの内容をオリジナル媒体の非認可

コピーであると疑わしい記録媒体から再生された修正信号ストリームの内容と比較して再生情報を抽出する工程を備えている請求項16記載の方法。

【請求項18】 上記再生情報は記録媒体識別、再生装置識別及びデータストリーム再生時のうちの少なくとも一つを含んでいる請求項16記載の方法。

【請求項19】 修正用のメッセージ領域としてフレームの少なくとも一つの領域を選択し、該選択された領域を修正する工程を備えている請求項16記載の方法。

【請求項20】 上記選択は画像属性に基づいて実行される請求項19記載の方法。

【請求項21】 上記画像属性は画像境界線である請求項20記載の方法。

【請求項22】 上記選択されたフレーム領域を上記再生情報の要素を指定する修正フレーム領域に置換する工程を備えている請求項19記載の方法。

【請求項23】 上記選択されたフレームを修正フレーム領域への置換の前に調整する工程を備えている請求項22記載の方法。

【請求項24】 連続するフレームから修正用に様々な領域を選択する工程を備えている請求項19記載の方法。

【請求項25】 上記領域は画素ブロックまたはマクロブロックを備えている請求項19記載の方法。

【請求項26】 上記再生情報をMPEG信号ストリームに埋め込む前に上記画素ブロックまたはマクロブロックをスクランブル化する工程を備えている請求項25記載の方法。

【請求項27】 上記MPEG信号ストリームを修正する前に上記再生情報を暗号化する工程を備えている請求項16記載の方法。

【請求項28】 上記MPEG信号ストリームを修正する前に上記再生情報を符号化してスペクトラム拡散キャリアを生成する工程を備えている請求項16記載の方法。

【請求項29】 上記MPEGデータストリームのBフレームのみが置換用フレーム領域として選択される請求項22記載の方法。

【請求項30】 上記MPEG信号ストリームを修正する前に上記再生情報の誤りを訂正する工程を備えている請求項16記載の方法。

【請求項31】 オリジナル記録媒体から再生されたMPEG符号化ビデオ信号ストリーム及び／または音声信号ストリームの形で圧縮デジタル情報を処理して上記オリジナル媒体からの信号ストリームの非認可コピーを追跡可能にする装置であって、

上記圧縮信号ストリームを含むオリジナル媒体の再生の出所に関するコピー源メッセージを得る検出器と、

上記オリジナル記録媒体からの再生時に上記MPEG信号ストリームを受け取るレシーバと、

上記コピー源メッセージを上記MPEG信号ストリームに埋め込むコンパイナとを備えている装置。

【請求項32】 上記MPEG符号化信号ストリームの選択されたビデオフレーム内の画素ブロックの位置を格納するメモリを備えている請求項31記載の装置。

【請求項33】 上記検出器により受け取られたコピー源メッセージをスクランブル化するスクランブラを備えている請求項31記載の装置。

【請求項34】 上記検出器により受け取られたコピー源メッセージを暗号化する暗号化部を備えている請求項31記載の装置。

【請求項35】 上記コピー源メッセージを上記MPEG信号ストリームに埋め込む前に上記コピー源メッセージを符号化してスペクトラム拡散キャリアを生成するエンコーダを備えている請求項31記載の装置。

【請求項36】 元のデータストリームからコピーされたデータストリームを圧縮デジタル信号の形で処理してコピーの出所に対する追跡を可能にする方法であって、

上記元のデータストリームを受け取る工程と、

所定サイズのメッセージホールを上記元のデータストリーム内の予め定義された不規則に発生する位置に配置する工程と、

上記メッセージホールに埋め込まれたコピー源メッセージを上記データストリ

ームから抽出する工程とを備えている方法。

【請求項37】 記憶媒体に記録されるデータ含有デジタル信号を処理して上記データの非認可コピーをコピー者まで追跡できるようにする方法であって

、
上記デジタル信号ストリームが格納された媒体から上記ストリームを再生することによりデジタル信号ストリーム含有データを受け取る工程と、

上記データの再生の出所に関するコピー源メッセージを得る工程と、

上記コピー源メッセージを含んだスペクトラム拡散キャリアを生成する工程と

、
上記符号化されたコピー源メッセージを上記デジタル信号ストリームと結合する工程とを備えている方法。

【請求項38】 上記デジタル信号ストリームが圧縮される請求項37記載の方法。

【請求項39】 上記デジタル信号ストリームがMP E G圧縮標準形式で符号化される請求項38記載の方法。

【発明の詳細な説明】

【0001】

技術分野

本発明は一般に、情報一関連媒体（例えばDVD）等のソースから得たデータのためのコピー保護に関し、さらに詳しくは、無許可でコピーする者が確認できるようにするためのデータ修正に関する。

【0002】

背景技術

映画、ソフトウェア等の記録されたプログラム資料の録画、及び再生のための消費者用装置は広く入手可能である。そのような資料に関する媒体は、磁気テープ、VCRを使用した録画／再生を含み、また最近では、CD、CD-ROMやDVDといった光学媒体もある。DVDは、新しい光学ディスク技術であり、各層で一般に133分の映画のための十分な情報を保持できる。DVDは、ビデオ資料を効率よく記憶し、目に見えるような劣化なく再生できる、いわゆるMPEG-2データ圧縮基準をしばしば使用している。ソフトウェアで実施しているビデオや多種の情報のための他の基準も利用できる。

【0003】

娯楽映画等のプログラム資料の製造コストはかなりのものであるが、コピーする者のコストは比較的安い。したがって、映画の著作権侵害が映画産業の収入源の重大な損失となっている。実際、コピーする者、主に不法ビデオカセットによって年間20億ドル以上を損失している。DVDプレーヤからのオーディオ、ビデオプログラムには高度な忠実性があり、VHSからのものよりもかなり多くのコンテンツを保持できるので、DVDはこのような状況をよりひどくしている。

【0004】

著作権侵害を妨害する2つの試みが映画の合法製作者と商業価値のある他の業種によって開発されており、そのソフトウェアとは：（1）再生時にある方法で資料をひずませる信号を保護された媒体に注入するもの、及び（2）可視、又は不可視の静止符号、又は元の資料のソースを確認するマークを注入して隙間を入れるものである。

【0005】

隙間は、一般に保護された資料を書く時点で媒体に入れるものである。例えば、情報一関連媒体が製作される郵便局や他の施設である。しかし、一度記録すると、透かしは固定されてしまう。隙間は資料を書くソースを確認できるが、無許可でコピーする者の確認に関する情報はない。コピーする者の確認は再生、コピーする時点で収集された情報から得られなくてはならない。これは、コピーするものが使用する装置と、コピーする時間の確認としての情報を含む。しかし、再生ユニットでこの情報を提供する電気回路は、ユニットのコストを吊り上げてしまい、競合産業において深刻な欠点となる。本発明の目的は、使用する時点で、保護資料のコピーするものによって変化し、保護資料にとって唯一の追跡信号を生成することである。本発明の他の目的は、再生ユニットのコストをさほど上げることなくそうすることである。本発明のさらなる目的は、媒体のドメインによって、デジタル又はアナログスペースでこの機能を果たすことである。

【0006】

予測可能性が信号をソースの偽造指定をするようフィルターで取り除く、又は変えることができるようにしてしまうことになるため、保護信号はさらに予測できなければならない。さらに、信号ストリームに注入されるコピーメッセージのソースを重い送信ノイズや、意図しない又は意図的なひずみのある環境で復元しなければならないため、保護システムは頑丈でなければならない。さらに、送信メッセージは、検出、改変、除去しにくくなくてはならない。メッセージのコンテンツは、送信ビット認識ソースがあっても、保護され、復号しにくくなくてはならない。最後に、出力ストリームは、適切なドメインで正当に復号できるよう合法でなければならない。本発明のさらなる目的は、保護された媒体の信号ストリームに注入した、コピーする者が検出又は変更しにくい使用信号のポイントを生成することにある。さらに使用信号のポイントは、ビデオに加えてオーディオやソフトウェアを含む保護するあらゆる種類のコンテンツに注入できるようにする。

【0007】

発明の開示

本発明によれば、情報一関連圧縮デジタル信号ストリームを処理する方法は、“コピーのソースメッセージ”をビデオフレームのピクセルブロック又はマクロブロックに配分された“ランニングマーク”の形式で加えることによりストリームを改変し、無許可コピーのソースを追跡できるようにするものである。正常使用における改変信号ストリームは、使用者が見るためのテレビ表示装置又はモニタに送出される。ランニングマークは薄く、普通は目に見えない。しかし、信号ストリームを無許可で録画するために使用する場合は、記録にはコピーのソースメッセージと共に元の圧縮デジタルストリームが与えられる。したがって、元のデジタル信号ストリームと改変信号ストリームを比較すれば、メッセージが抽出でき、コピーした者を追跡できる。

【0008】

好ましい実施例では、例えば、圧縮デジタル信号ストリームは、MPEG基準で符号化されたオーディオ/ビデオデータストリームである。ピクセルブロック、又はマクロブロックは、コピーメッセージのソースを運ぶための“メッセージホール”という候補領域としてフレームベースで選択される。メッセージホールは、ビット調整で変更可能な画像フレームのブロック又はマクロブロックに関するMPEGビットストリームの選択されたセグメントに対応している。

【0009】

コピーのソースメッセージは、予め記録された資料（映画等）の再生時に置換ビットとして信号ストリームに与えられる。メッセージホールとなるよう選択されたマクロブロックは様々な方法でビット調整される。好ましい実施例では、例えば、調整は、好ましくは(DCT) ディスクリートコサイン変換形式ドメイン（あるいはビデオドメイン）で規定量のノイズを選択されたマクロブロックに加算、又は選択されたマクロブロックから減算して行う。これにはマクロブロック内に歪を広げ、ピクセル歪を見えにくくするという利点がある。

【0010】

メッセージホールは、フレーム内にランダムに位置しているが、使用者には見えない又は殆ど見えない改質を調節する、画像境界等の画像の織目部分に存在するよう選択されることが好ましい。候補メッセージホールの位置は、オペレータ

の入力又は機械操作で、例えばフレームごと（又は一度にマルチプルフレーム）に選択してもよい。選択された位置は再分布されて、フレーム内でより均一なメッセージホールの分布となる。再分布はメッセージホールの数をフレームごとのメッセージホールの目標数に近づけるよう保持することに基づき、1フレーム又は少数の隣接するフレーム内で密集しないようにする。

【0011】

行内の選択されたマクロブロックに含まれるビット数を相対的に一定に保つために、ランニングマークを付加した後に、ビット数をMPEG基準に一致させ、バッファの底流又は流出を防ぐためにMPEG再符号化をして補間してもよい。これは、例えば、マクロブロックビットの元の数とランニングマークで符号化した後のビット数が略同じになるまで、マクロブロックピクセルの量子化スケール要素を変更すればできる。

【0012】

メッセージホール候補の選択に続いて、再符号化処理中に、交互メッセージホールが規定の基準を満たしているか、コピーメッセージビットのソースを挿入するために使用すべきかを決定するために、候補メッセージホールをテストし、両方を受け入れ、放棄し、再分布する。不適切な、又はおそらく不適切であろうメッセージホールとしては、例えば：DVD区分境界にある行のメッセージホール、ビット補間で再符号化できない行のメッセージホール、確実に復号できないメッセージビットを含むメッセージホール、及び再符号化に続いてスキップされるメッセージホールがある。

【0013】

コピーのソースメッセージは、以下の1つ以上に対応している：無許可でコピーをする元の許可再生ユニットの連続番号、コピーする媒体の連続番号、及びコピーの時間。この情報は、再生ユニットと記録媒体を提供する機関によって保持される記録を使用して、無許可でコピーする者を確認する助けとなる。

【0014】

無許可でコピーする者がその情報をフィルターで除去、又は改変する目的でコピーのソースメッセージを復号するのを防ぐ助けをするために、メッセージは記

号化し、混乱させるのが好ましい。コピーする者の追跡を困難にする多数の再生ユニットで生成したコピーのソースメッセージビットを相関関係検出できるコード分割マルチプルアクセス符号化(CDMA)を使用してメッセージを符号化すると便利である。

【0015】

メッセージホールのランニングマークコンテンツを抽出し、復号化するためには、各メッセージブロックの位置をメモリに記憶させる。関連媒体は、メッセージホールの位置と共に元のプログラム資料に付けて生成する。コピーのソースを確認するためには、テストで媒体と同期して関連媒体を再生する。対応する位置でテストで媒体から読まれ、関連媒体からも読まれたピクセルブロックをお互いに比較し、コピーのソースメッセージビットで改変された元の画像かどうか判断する。例えば、逆でもよいが、メッセージホールでの画像改変は、“1”、改変がないものは“0”と指定する。2進法フォーマット以外を含む他の情報フォーマットは選択的に実施してもよい。

【0016】

本発明の特異な目的、利点、新規の特徴は、以下に続く説明の部分で述べ、ある部分では、当業者が以下を検査すれば明らかとなり、本発明の実施によってわかるであろう。本発明の目的及び利点は、添付のクレームで特に指摘した手段、及び組合せによって達成できることがわかる。

【0017】

図面の簡単な説明

図面について説明するが、明細書中、同様の符号を付した構成要素は同様の構成要素を表し：

図1は、本発明によるMPEGランニングマーク位置の作成を示す高レベル機能ブロック図である。

【0018】

図2は、コピーのソースメッセージを行う3つのメッセージホールのある画像を示す。

【0019】

図3は、ビデオストリームで実施したメッセージホールを示す線図である。

【0020】

図4(a)、及び4(b)は、本発明によるランニングマークシステム符号器及び復号器を示すブロック線図である。

【0021】

図5は、データ記号基準を使用したメッセージビットの記号化を示す線図である。

【0022】

図6(a)、及び6(b)は、鎖状誤差訂正符号化及び複合化の線図である。

【0023】

図7は、本発明の実施例で実行する包旋状誤差訂正コードを示す。

【0024】

図8は、本発明の実施例で実行するCDMA符号化処理を示す線図である。

【0025】

図9(a) - (e)は、ビデオストリームで実施したコピーのソースメッセージのセキュリティを向上させるためのマルチプル可変長配行を示す。

【0026】

図10(a) - (d)は、本発明によるビット符号化の変形例を示す

図11は、本発明に一形態でのDCTキャリア係数へのノイズ組成を加えることによる元の信号波形の広がりを示す。

【0027】

図12は、本発明におけるRMカメラ生成のためのDCTの試みを示す擬似コードである。

【0028】

図13は、DCTの試みを使用した輝度ブロックのためのピクセルドメインにおけるノイズパターンを示す。

【0029】

図14(a) - (d)は、ランニングマークデータストリームのレイアウトと構成の線図である。

【0030】

図15は、本発明によるDVD信号ストリームで行われるビット抽出を示す線図である。

【0031】

図16は、本発明によるランニングマーク生成の動作を示すアルゴリズムである。

【0032】

図17、及び18は、本発明の一形態によるDivxマーク挿入前、挿入後のビデオグループのレイアウトを示す。

【0033】

図19は、gMH行を定義するアルゴリズムである。

【0034】

図20は、本発明に一形態によるマクロブロックの分類のための擬似コードである。

【0035】

図21は、分類機能によって要求される初期化（）のための擬似コードである。

【0036】

図22は、マクロブロック分類の他の実施例のための擬似コードである。

【0037】

図23は、処理する典型的なフレーム区分の線図である。

【0038】

図24は、再分配手順の擬似コードである。

【0039】

図25は、再分配機能によって要求される初期化（）のための擬似コードである。

【0040】

図26は、メッセージホールの予備ビットの補間するための流れ行におけるマクロブロックのためのMquantの増加数値を示す。

【0041】

図27は、本発明によるMPEG再符号化の第一形態を示すアルゴリズムである。

【0042】

図28は、第2形態を示すアルゴリズムである。

【0043】

図29は、 k 回 ($k = (d + 2)$) 繰り返すための j の位置決めアルゴリズムである。

【0044】

図30は、MPEG再符号化の第3形態を示すアルゴリズムである。

【0045】

図31は、本発明によって行われる機能をまとめた高レベル線図である。

【0046】

図32は、GenRMプログラムを示すアルゴリズムである。

【0047】

図33は、見る角度を変えたビデオ表示を示す。

【0048】

表の簡単な説明

表1は、 $n = 8$ の場合のアダマールーウォルシュ波形を示す。

【0049】

表2は、列と行の配列後のアダマールーウォルシュ波形を示す。

【0050】

表3は、別に規定したアプリケーションのプロバイダ決定ストリームのための `DivxStreamId` を示す。

【0051】

表4は、RMGヘッダのデータ構造を示す。

【0052】

表5は、RMGデータのデータ構造を示す。

【0053】

表6は、b c l n c rモードからb c l n c r値を得るためのルックアップ表である。

【0054】

表7は、再符号化の繰り返し状態を示す。

【0055】

表8は、他のメッセージホール処理モードを示す。

【0056】

発明の詳細な説明

1. 概要

この発明は、仮想的に不可視なマークの圧縮されたデータストリームへの付加を提供する。このマークはDVDなどの情報記録媒体の公認されていないコピーのソースの追跡を可能にする「ソースオブコピーイングメッセージ」を構成する。本発明の最良な形態はMP E Gの分野で生成されるビデオストリームの内容の下に述べられるが、本発明はこれに限定されるものではない。例えば、ソースオブコピーイングメッセージは、同様の原理を用いるオーディオ信号やソフトウェアにも組み合わせることができる。これらのマークは、ビューアでは見ることができず、オーソライズされた機関によって維持されるハードウェアによって検出され、デコードされる。より詳しくは、メッセージは特定の再生装置およびDVDのような特定の原本記録媒体、および再生時刻のシリアルナンバーを特定する。明らかなように、このメッセージは再生装置によって媒体が再生されるごとに一意的に生成される。媒体（公認されたコピーから海賊版が作られたものであるとしても）を再生し、ビデオストリームに付加されたマークに埋めこまれたメッセージをデコードすることによって、機関は複製者の追跡を開始することができる。これらのマークの内容は再生されるたびに変わるので、このマークは「ランニングマーク」またはRMと称され、著作の時点で一旦作成されたら変化しないすかしとは異なる。

【0057】

図1を参照して概要で述べられるのは本発明のランニングマークシステムが実施される環境である。記録スタジオでは、ムービーなどのプログラム素材を運ぶ

マスターデジタルリニアテープDLTが、ユニット124によって、DLTの内容をMPEGビデオストリームに変換する処理がなされる。次に、プログラム素材は、DVDマルチプレクサユニット126によってDVD記録に適したフォーマットに配置される。エンコーダ124およびマルチプレクサ126は、この業界で標準的な装置を含んでいる。DVDマルチプレクサ126は、MPEGビデオストリームに、データエントリのためにビットストリームに空領域を予約するサブピクチャ箇所保持部を付加する。

【0058】

マルチプレクサユニット126の出力においてビデオストリームによって表されるイメージは、ディスクフォーマッティングユニット128によって、ビデオストリームのフレームの中でランニングマークの位置の候補を特定するために処理され、ビデオを保持するDVDは、ユニット130でスタンプされ、シリアル化される。ディスクフォーマッティングユニット128は、マークが見えないかまたはじゃまにならない画像のロケーションにランニングマークを埋めこむために、各ビデオフレームから候補ブロックまたはマクロブロックを探す。MPEGビデオストリームのこれらのロケーションで「メッセージホール」（メッセージホールは、本発明では、それぞれビットモジュレーションに変換され得るピクチャフレームのブロックまたはマクロブロックに対応するMPEGセグメントである。）を表すデータビットには、フレーム中のロケーションを特定するためのアドレスオフセットが与えられる。置換されたブロックは、ディスクフォーマッティングユニット128によって、プレースホルダに保存される。

【0059】

ユニット128は、ランニングマーク保持ユニット132（好ましくはテープまたはDVD）に、オリジナルのMPEGストリームだけでなく、再生装置、ディスク、および再生時刻のためのビデオストリーム情報を運ぶメッセージホールロケーションを送る。保持ユニット132は、複製者が追跡されるのを可能にするのを助けるために、テストの下記録媒体からランニングマークを抽出しデコードするのに用いられるために、公認された機関に参照媒体として保持される。

【0060】

ユーザは、機関から発行されたシリアルナンバーによって特定されるプレイバックユニット134が提供される。プレイバックユニットが支給されたユーザは、機関に登録される。また、登録ユーザによって閲覧されるためにムービーを運ぶ事前に記録された媒体も、機関によって支給されシリアルライズされる。プレイバックユニットおよび事前に記録された媒体の登録ナンバーは、不可視であるがプレイバックユニット134によって読み取り可能に保存される。例えば、プレイバックユニットのシリアルナンバーは、ユニット内のリードオンリーメモリに保存され、媒体のシリアルナンバーは、好ましくは媒体に記録されているものの中にエンコードされる。プレイバックユニット134は、都合よくは媒体が再生される時刻を特定するリアルタイムクロックを含む。これらの情報コンポーネントは、媒体が再生されるごとに、プレイバックユニット134によって、シリアルライズされ、生成されるランニングマーク中にエンコードされる。

【0061】

登録されたプレイバックユニット134は、事前に記録された媒体の内容を再生し、通常は、ビデオストリームがテレビ画面やモニタ135に再現される。一例は、事前に記録されたムービーを見る目的でDVDを借り受けまたは購入するカスタマである。ビデオストリームに埋めこまれたランニングマークは、モニタ135上では見ることはできず、ビューアは、ランニングマークを関知したり影響を受けたりしない。しかし、もし媒体がリプロダクションユニット136を用いてコピーされた場合には、そのようにして作成された公認されていないコピー138は、オリジナルのビデオストリーム（ムービー）だけでなく、プレイバックユニット134、機関から発行されたオリジナルディスク、および再生時刻を特定するランニングマークをも含む。公認されていないコピーは、マーク抽出システム122によって再生され、複製者を追跡するために好ましくは機関によって保持される。マーク抽出システム122は、マーク保持システム132から得られたブロックロケーションでのランニングマークの内容を読み取る。ランニングマークの内容の適切なデコーディングは、公認されていない複製の出所の追跡に用いられる。

【0062】

ランニングマークは、好ましくは選択されたフレーム内のビデオストリーム中に戦略的に分配されたピクセルのブロックを含むが、ランニングマークは、ビューアによる関知を阻止するために、連続するフレームないの同じロケーションには現れない。フレーム中のランニングマークの位置は、プレースホルダセクタによって定義されたメッセージホールロケーションに対応する。マーク自信は、メッセージホールロケーションのピクセルがオリジナル画像の内容から変更されるか否かによらず、“1”または“0”の論理で表されてもよい。この基準は、反転され、または他の異なるフォーマット、例えばバイナリデータではなく4値などにエンコードされ得る。一例としては、もし、メッセージホールが、ブロックのオリジナルの画像内容から変更されたピクセルのブロックを含む場合には、そのメッセージホールは論理“1”としてデコードされ、もしメッセージホールロケーションに対応するピクセルブロックが、そのロケーションのオリジナル画像と比較して変更されていない場合にはメッセージのないようは論理“0”としてデコードされてもよい。これらの論理1および0は、フレーム内で、そして、フレームごとに、コピーのソースに対応する完全なメッセージを組み立てるために、連続して読み込まれる。

【0063】

図2は、オペレータによる手動で、または既に述べた画像属性の基準に従った装置によって自動的に、ビューアに対しては不可視にフレーム中に配置されたメッセージホールロケーションa、b、およびcを含むあるビデオフレームに対応する画像を表す。例えば示された画像の境界などの変わり目、または加工された領域は、選択に好ましい。もし、ブロックaの内容がオリジナルの画像から変更されると、それは値が“1”のランニングマークとして、さもなくば、もしフレーム中のマークaがオリジナル画像と同一である場合には値が“0”の値のランニングマークとしてデコードされる。ここで重要なことは、メッセージホールのロケーションは保存され、ランニングマークを表すそれらのロケーションのピクセルブロックの内容は、評価のために、ランニングマークが変更された画像を表すか否かを決定するために同じロケーションのオリジナル画像と比較される。しかしながら、他のエンコード基準、例えば、バイナリまたは他のタイプのデータ

を指定するための2またはそれ以上の異なって変化したランニングマークを使うことも、ランニングマークの値を決定するのに用い得ることは理解されるべきである。

【0064】

図3は、MPEGビットストリーム中のメッセージホールを表す。いくつかの参照ロケーションに関しての各メッセージホール、ビットサイズ、およびロケーションオフセットは、記録媒体のサブピクチャチャンネルに保存される。複製のソースに対応するMPEGビットは、プリエンコードされ、サブピクチャチャンネル内に保存される。プレイバックユニット134でのランニングマーク(RM)データのエンコーディング過程では、プレイバックユニットおよび媒体のシリアルナンバー、ならびに再生時刻が読み出され、入力されるMPEGビットストリームのメッセージホール中のビットが、この複製情報のソースに対応するMPEGビットに置き換えられる。いかなる数のビットも、機関によって発行されたプレイバックユニットおよび媒体の数に適応して適切に定義された範囲で、各メッセージエレメントに対して指定されることができる。これらのエレメントに対応したビットは、媒体が再生されるごとに再生ソースを特定するビットストリームを形成するためにシリアルライズされる。

【0065】

2. ランニングマークシステム

上記概要のように、ランニングマークを用いてビデオストリームに埋めこまれたメッセージは、好ましくは1またはそれ以上の次のフィールド(1)プレイバックユニットのシリアルナンバー、(2)記録媒体(例えばDVD)のシリアルナンバー、および(3)ソースの複製を示す再生時刻を含む。各ランニングマークメッセージは規定された数、例えば、連結され、複製のソースオブコピーイング特定フィールドを含む場合には、128のビットから成り立っている。

【0066】

特定のためのソースオブアイデンティフィケーションメッセージがエンコードされ、MPEGビデオストリームに挿入される方法を図4aに示す。記録媒体が、閲覧されるため(または公認されていない複製のため)に再生される際に、プ

レイバックユニット134 (図1) でリアルタイムに生成されるメッセージ、図4aにおいてはMsglnと称される、は、メッセージエンコーダ302で適用される。このエンコーダ302は、メッセージの全てのフィールドを各フィールド (プレイバックユニットシリアルナンバー、ディスクシリアルナンバー、再生時刻) に割り当てられたビットの数にしたがってビットストリームBaに変換する。プレイバックユニットのシリアルナンバーは変更されないで、好ましくはプレイバックユニット内のリードオンリーメモリ (ROM) から読み出される。ディスクのシリアルナンバーは、再生中に直接ディスクから読み込まれる。再生時刻は、プレイバックユニット内のシステムクロックから得られる。全てのビットは、ビットストリームBaを形成するために連結される。

【0067】

好ましくは複製のソースオブコピーイングメッセージは、DVDまたは他の媒体のビデオストリームに複数回連続して挿入されてもよい。これは、DVDの中でメッセージデータに利用可能なフレームの数が、コピーソースを特定するために必要なデータの記憶要求よりもはるかに大きいために可能となる。メッセージの安全性を高めるのを助けるためには、フォーマットまたは各挿入の内容を異ならせてもよい。例えば、連続するいくつかの各エントリのコーディングや、同じ特定を表す全ての情報は、機関のデコーダによって理解される異なるフォーマットで格納されてもよい。

【0068】

最良の形態では、唯一のデータのビットが各メッセージホールに埋め込まれる。しかし、発明の範囲内で種々の変形が可能である。例えば、メッセージの2またはそれ以上のビットが、何らかの適切な方法でエンコードされて一つのメッセージホールにもたらされてもよい。また、1つのメッセージビットが2またはそれ以上のメッセージホールに分配されてもよい。

【0069】

さらに他の変形例としては、ランニングマークは、MPEGビットストリーム中よりもビデオストリーム中に直接埋め込まれてもよい。すなわち、フレームの各ピクセルまたは選択されたいくつかのピクセルが、直接、メッセージビットと

ともにエンコードされてもよい。これは、例えば、オリジナルに関してピクセルの輝度を変化させて、所定の論理値を示すようにしてもよい。この技術の現実的な実施は、論理値1または0をそれぞれ表すために、各輝度が増加または減少されるピクセルによって示されるデータのPNシーケンスによって提供される。

【0070】

2. 1 メッセージの暗号化

ビットストリームBaが暗号化ユニット304に供給される。暗号化ユニット304では、再生源となるメッセージを保護するため、また、偽造されたものであるけれども適法なメッセージを攻撃者が作成してシステムの信用を落とすのを阻むために強固な暗号化が行われる。メッセージのビットストリームBaの暗号化には、図5に示すような5重の暗号化が適用されることが好ましい。図5に示す5重の暗号化ではDES (Data Encryption Standard) アルゴリズムを使用している。5重のDESアルゴリズムの前と後には128ビットの置換が行われる。まず、128ビット置換器Ps0 320によってビットストリームBaがスクランブルされ、スクランブルされた結果が左半分の部分Ba1と右半分の部分Ba2とに分割される。

【0071】

左半分・右半分の各部分は64ビットで構成される。左半分・右半分の各部分はそれぞれ、DESによって鍵K0-K4を用いて5回暗号化される。鍵K0-K4の各々は64ビットの長さを有する。その後、置換器Ps1 322によって、左半分の部分と右半分の部分とが結合され、結合された結果がスクランブルされ、128ビットのビットストリームBbが生成され、これが出力される。

【0072】

DESに代わる暗号化の手段としては、公開鍵暗号系がある。公開鍵暗号系は、整数論、有限体構造および代数的な符号理論に基づいている。公開鍵暗号系の例としては、RSA、McEliece、ElGamal、楕円曲線アルゴリズムなどがある。IDEA (International Data Encryption Algorithm) と呼ばれる新しい暗号アルゴリズムも暗号化手段のよい候補である。IDEAは、1992年にXuejua LaiとJam

es Masseyとによって提案された暗号化アルゴリズムである。

【0073】

2. 2 誤り訂正

暗号化ユニット304から出力された128ビットのビットストリームBbは、テープまたはディスク再生技術において不可避なひずみを訂正するために誤り訂正符号化器306に供給される。ひずみは、デジタル-アナログ変換処理、ソースレコーディング、ディスクの製造、テープまたはディスクの複製などの過程で生じる。模倣者は、ビデオソース(video source)を故意にひずませることによって本発明のランニングマークシステム(the running mark system)を攻撃するかもしれない。埋め込まれたランニングマーク(running mark)を確実に回復することができるようにするために、強固な誤り訂正符号化(ECC)が行われることが好ましい。さまざまなECCアルゴリズムおよびそれらの組合せを使用することができる。高い誤り訂正能力を達成しかつ少ない計算量で符号化するためには、図6aおよび図6bに示すような接続ECCシステムを適用することが望ましい。このECCシステムでは、8ビットシンボルの(32, 16)縮小リード・ソロモン符号化器によって128ビットの入力ビットストリームBbが符号化されて256ビットのビットストリングBc1が得られる。得られたビットストリームは、図6aに示すように符号化率1/2、拘束長K=7の畳み込み符号化器によって符号化されて528ビットの出力ビットストリームBcが生成される。この符号化処理では32シンボルのうち1つの符号語だけしか存在しないため、速度が速く、符号化器の間にビットインタリーブが必要とされない。

【0074】

リード・ソロモン(RS)符号は一般化BCH符号のある特別のサブクラスである。ガロア体GF(2^m)において定義された通常の(n, k)RS符号は、長さn($=2^m-1$)の符号語のシンボルを有する。ここで、mは正の整数である。各シンボルはmビットで構成される。この符号の最小距離は(n-k+1)であり、この符号は、最大限 $t = \lfloor (n-k)/2 \rfloor$ 個の誤りまで訂正できる。ただし、 $\lfloor x \rfloor$ は、x以下の最大整数を示す。パリティ検査シンボルの数は(n

$-k)$ である。この符号の生成多項式 $g(x)$ は以下の示すとおりである。

【0075】

$$g(x) = (x - \alpha)(x - \alpha^2)(x - \alpha^3) \cdots (x - \alpha^{2l})$$

α は $GF(2^m)$ の原始元である。 $m = (m_0, m_1, \dots, m_{k-1})$ は、一致多項式 $m(x)$ を伴う k シンボルのメッセージブロックであるとする。ここで、 $m(x) = m_0 + m_1x + m_{k-1}x^{k-1}$ である。また、 $c(x) = (C_0, C_1, \dots, C_{n-1})$ は、一致多項式 $c(x)$ を伴う n シンボルの RS 符号である。そして、ある(組織) RS 符号化アルゴリズムは以下の2つのステップからなる。

【0076】

ステップ1: $m(x)$ と x^{n-k} との積を $g(x)$ で割る。 $d(x) = d_0 + d_1x + d_{n-k-1}x^{n-k-1}$ を別途記憶しておく。

ステップ2: $C(x) = x^{n-k}m(x) - d(x)$ をセットする。ここで、 $c = (-d_0, -d_1, \dots, -d_{n-k-1}, m_0, m_1, \dots, m_{k-1})$ である。

【0077】

上述の多項式の計算は $GF(2^m)$ 内で行われる。シフトレジスタを用いることによって効果的に行うことができる。CDやDVDプレーヤの多くのアプリケーションにECCが含まれている。許容される符号長は、通常の長さである n ($= 2m - 1$) シンボルよりも短いため、縮小RS符号を用いることができる。 S は、 (n, k) RS符号の符号語のサブセットであるものとする。 (n, k) RS符号の符号語中もっとも右の j 座標は値0を有する。そうすると、 S 内のすべての語からもっとも右の j 座標を削除することによって $(n - j, k - j)$ 縮小RS符号が形成される。元のRS符号を生成する同様のシフトレジスタ符号化器を、短縮RS符号を生成するために使用することができる。もし削除された座標を消去された位置として取り扱ったならば、オリジナルのシフトレジスタ符号化器と同じものを消去したものの復号化とともに復号のために使用することができる。(消去したものは、受信したシンボルの値が疑わしいことを示すシンボルである。)これに代えて、この発明では、削除された座標をゼロで置き換える手法を使用する。

【0078】

この発明のRMシステムでは、128ビットのメッセージを符号化するためにGF(2⁸)における短縮RS(32, 16)符号が使用される。各シンボルは8ビットの長さを有する。この符号では最大t=8のシンボルの誤りまで訂正することができる。原始元αは、原始多項式P(x)の根である。ここで、P(x)=1+x²+x³+x⁴+x⁸である。RS符号を効果的に復号するために、Berlekamp-Masseyシフトレジスタアルゴリズムを使用することができる。

【0079】

畳み込み符号化器は入力ビットストリームを多数のインパルス応答とともに畳み込んでECCビットを生成する。図7は、符号化率1/2、拘束長Kの畳み込み符号化器を示す図である。ここで、x=(x₀, x₁, x₂, ...)は入力ビットストリーム、(g⁽⁰⁾, g⁽¹⁾)は符号化器に対する2つのインパルス応答であるとする。そうすると、符号化されたビットストリームy=(y₀⁽⁰⁾, y₀⁽¹⁾, y₁⁽⁰⁾, y₁⁽¹⁾, y₂⁽⁰⁾, y₂⁽¹⁾, ...)が以下に示す離散的な畳み込みによって得られる。

【0080】

$$y_i(j) = \sum_{l=0}^{K-1} x_{i-lg_l}(j)$$

ここで、j=0, 1である。畳み込み符合を復号するために、ビタビ復号化器を使用することができる。連接誤り訂正システムのために、畳み込み符号化器は拘束長K=7および符号化率1/2を有する。インパルス応答は、g⁽⁰⁾=(1011011)およびg⁽¹⁾=(1111001)である。

【0081】

2.3 スペクトル拡散符号化/CDMA

多数の再生ユニットからのビデオ出力を結合することによってこの発明のランニングマーク(running mark)システムを攻撃することを模倣者は試みるかもしれない。この方法では、埋め込まれたランニングマーク(running mark)は削除され、コピー元のメッセージを回復させることが困難

になるかもしれない。このため、この発明の1つの局面によると、符号分割多元接続(CDMA)符号化ユニット308を設けて、再生ユニットによって生成されたランニングマーク(running mark)の復号化の分離をたとえノイズが存在しているときであってても可能にしている。CDMAは、指定された周波数内で多数の伝送チャンネルを供給するために特別な符号を使用する変調の一種である。Hewlett-Packard Journal, December 1993のページ90-97「North American Cellular CDMA」というタイトルの記事にCDMAの標準について述べられている。以下の説明はこれを参考にしている。ランニングマーク(running mark)を符号化するためにCDMAを使用することの利点は、特性に基づいている。種々の再生ユニットは種々の相関が低い波形を用いてビットを符号化してMPEGビデオストリームに加えるという特性に基づいている。復号の間、受信信号と関連波形との内積をとることによって再生ユニットの送出ビットは回復される。これらの波形の間の相関は低いため、たとえ種々の再生ユニットからの出力波形の結合として受信波形が生成されていても内積をとることによって送出ビットを直接取り出すことができる。

【0082】

この発明の1つの局面に従うと、CDMA符号化においては擬似ランダム雑音(PN)系列が使用されることが望ましい。これらPN系列はスペクトル拡散多重アクセス用アプリケーションに使用されるものであり、時間領域においてランダム雑音のように振る舞う。また、周波数領域において信号を隠したり干渉を除去したりする特性が良好である。使用することができる別の種類の波形としては、バイナリーの直交波形の集合がある。直交波形が有効なのは、それらの相関が最も低いすなわち0であり、よりよくビットを取り出せるからである。この目的のためのバイナリー直交波形の望ましいタイプはHadamard-Walsh(HW)波形である。これはHWマトリクスの行を周知の方法で選択することによって得られる。

【0083】

しかし、非常に多数の再生ユニットには独自の波形が割り当てられる必要があ

るであろう。もし、各再生ユニットが異なった波形を使用したならば、要求されるHW波形の長さは非実用的なものとなる。また、未知の波形でビットを取り出すのは不可能という問題がある。もし多数の再生ユニットが共通の波形を共有したならば、模倣者はこれらの再生ユニットを使用してランニングマーク (running mark) を回復できないようなビデオソースを生成するかもしれない。この問題を解決するためにこの発明では、ランタイムの間、各ディスクプレーヤーに対して1つの波形を波形の集合からランダムに選択する。これは有効である。なぜなら、模倣者のすべてのディスクプレーヤーがランタイムの間同じ波形を選ぶ確率は低いからである。また、映像内のランニングマーク (running mark) の各コピーについて波形はランダムに選ばれる。そして各時間の間、同じまたは異なった映像が再生される。少数の異なる波形だけがあるため、ビットの取り出しプロセスではそれらのすべてを試して内積の最も高いものを選択することによって送出ビットを決めることができる。高速HW変換をこの目的のために利用することができる。

【0084】

もしどの波形を使用するかが決まっていれば、CDMAシステムに対する模倣者の攻撃を避けるために、CDMA波形は置換を使用してスクランブルされ、ビットストリームを送出するために1つの波形よりも多くの波形が使用される。直交特性を保存するために、波形はまず異なる列の間で置換され次いで異なる行の間で置換される。表1は、長さ8のHandmard-Walsh波形を示す。表2は、列および行の置換後のHandmard-Walsh波形を示す。

【0085】

【表1】

波形番号	値								2値再表現							
0	+1	+1	+1	+1	+1	+1	+1	+1	1	1	1	1	1	1	1	1
1	+1	-1	+1	-1	+1	-1	+1	-1	1	0	1	0	1	0	1	0
2	+1	+1	-1	-1	+1	+1	-1	-1	1	1	0	0	1	1	0	0
3	+1	-1	-1	+1	+1	-1	-1	+1	1	0	0	1	1	0	0	1
4	+1	+1	+1	+1	-1	-1	-1	-1	1	1	1	1	0	0	0	0
5	+1	-1	+1	-1	-1	+1	-1	+1	1	0	1	0	0	1	0	1
6	+1	+1	-1	-1	-1	-1	+1	+1	1	1	0	0	0	0	1	1
7	+1	-1	-1	+1	-1	+1	+1	-1	1	0	0	1	0	1	1	0

n=8の時のハダマード・ウォルシュ波形

【0086】
【表2】

波形番号	Pw1を用いた列並び換え	Pw1とPw2を用いた 列と行の並び換え
0	1 1 1 1 1 1 1 1	1 0 1 0 0 1 1 0
1	0 0 1 0 1 0 1 1	1 1 1 1 1 1 1 1
2	1 0 1 0 0 1 1 0	1 1 1 0 1 0 0 0
3	0 1 1 1 0 0 1 0	0 0 1 0 1 0 1 1
4	1 1 1 0 1 0 0 0	1 0 1 1 0 0 0 1
5	0 0 1 1 1 1 0 0	0 1 1 0 0 1 0 1
6	1 0 1 1 0 0 0 1	0 1 1 1 0 0 1 0
7	0 1 1 0 0 1 0 1	0 0 1 1 1 1 0 0

Pw1 0 1 2 3 4 5 6 7
2 0 4 1 6 5 7 3

Pw2 0 1 2 3 4 5 6 7
1 3 0 6 2 7 4 5

列及び行並び換え後のハダマード・カメルシエ波形

【0087】

スクランブルされたHW波形は、あらかじめ符号化されたディスクから読み出されまたはあらかじめROMに記憶されている。すなわち、ランタイム中に計算する必要がない。多数の波形を使用するためには、それらはビットストリームの同じビット位置について異なった再生ユニットに対する波形は異なっているように適切に選択されるべきである。

【0088】

図8を参照してCDMA符号化処理をより分かりやすく示す。まず、0から511の間の数がランダムに生成される。そして、列と行の置換を伴った264の連続的なHW波形が選択される。これらの波形とビットストリームBcの最初の半分の部分および2番目の半分の部分との排他的論理和のビット演算が行われ、 528×512 の長さの出力ビットストリームBdが生成される。バーストノイズをさけるために、ビットストリームBdのビットは列の順に形成される。

【0089】

2.4 スクランブル

より高い安全性をビットの送出に持たせるために、MVL P (multiple variable length permutation) 置換が使用されてCDMAビットストリームがスクランブルされる。図9aを参照して、置換器310において、種々の長さを有する多数の置換器を使用してCDMAビットストリームBdのビットが置換されて同じ長さのビットストリームBaが形成される。図9bに示すように、最大長256の8個の置換器を使用することができる。長さ7の置換器P0およびその逆置換器の例を図9cに示す。最終の置換を行うために、残っているビットの数は置換器の数よりも少ない。ギャップを満たすために”Don’t care”の語が詰め込まれる。置換の後、正常なビットは結果からスキャンされて送出される。逆置換処理の間、ユニット406のデコーダ400において、送出ビットが正しい位置に挿入された後、逆置換が実行される。図9dおよび図9eは、図9cに示した置換器を用いたこれらの処理の例を示す。上記置換は、上述したCDMA符号化処理と結合することができる。ビットストリームBeを送出する前に必要となる一時的な記憶スペースは、 528×512 ビットの代わりに256ビットである。置換されたビットストリームBeはビットモジュラ312に供給される。ビットモジュラ312は、入力されるMPEGビットストリームを修正することによって1またはそれ以上の送られてきたビットを最終ビデオ出力に埋め込む。デコーダ(図4b)内のビット復調器404において受信ビデオを加工することによってこれらのビットは回復される。

【0090】

2. 5 ビット変調及び復調

図4 aに示すように、ビット変調器312は、入力MPEGビットストリームのメッセージホールのビットストリームを変更することによってビットストリームBeの1個以上の送信されたビットを修正MPEGビットストリームBfに埋め込む。このビットストリームBfはビデオ信号を含む走行マークを出力するMPEG復調器314に付与される。各メッセージホールについて、そのサイズとある基準位置に対してオフセットされた位置とがDVDディスクの部分画像チャンネルに格納される。RMキャリアは送信用RMビットからなる(MBの画素値にも対応する)MH内のMPEGビットとして定義される。これらキャリアのMPEGビットは予め符号化され、部分画像チャンネル内に格納される。RM符号化時、MH情報とこれらキャリアとが読み出され、入力MPEGビットストリームのメッセージホール内のビットが送信用ビットに応じて上記キャリアに置換される。これらキャリアを格納する必要なデータ空間を削減するために、キャリアの一つがMPEGビットストリームのMHに予め格納される。MHは、その内部に2のn乗個の異なるRMキャリアが存在する場合に、nビットを発信することができる。最良の形態では、各MHにつき2個のキャリアが存在し、1個のキャリアのみが部分画像チャンネルに格納される。

【0091】

図4 bのビット復調器404は受け取ったビデオ信号を処理することでビットストリームBeを復元する。受け取られるビデオ信号はDVD再生装置などの他のソースや、VHSテープ再生装置やインターネットから得ることができる。走行マークを復号化する前に、ビデオディジタイザ装置402でNTSC復調などの適正な復調とフォーマット変換が実行される。ビデオディジタイザ402は高品質のA/D変換器を使用して入力ビデオ信号をデジタル化する。チャンネル補償器は、デジタル化されたビデオ信号を元のビデオ信号に合わせ、従来のフレーム位置合わせ技術とチャンネル等化技術とを使用してチャンネルの誤りを補償する。

【0092】

さらに図4 bに示すように、ビット復調器404は、ビデオフレーム内に配分

されたメッセージホールの元のビデオ信号とそのブロック位置またはマクロブロック位置とともに、ディジタル化されたビデオビットストリームB bを受け取る。ビット復調器404は、ディジタル化されたビデオストリームV bと元のビデオ（基準となるビデオと検査対象ビデオ）が同時に再生されている際にこれら二つのビデオを比較し、基準ビデオに記録された各メッセージホール位置により定義された検査対象ビデオのブロックまたはマクロブロックの内容を比較する。メッセージの位置に関して、元のビデオ（基準ビデオ）のブロックまたはマクロブロックと検査対象ビデオ内のブロックまたはマクロブロックが画像の変化を示していれば、ビット復調器は論理値1を生成し、元のビデオのブロックまたはマクロブロックと検査対象ビデオ内のメッセージホール位置のブロックまたはマクロブロックが同じであれば、ビット復調器404はビットストリームB eとして論理値0を生成する。このビットストリームB eは、復号器414でコピー源メッセージに変換される前に、ユニット406でデペルミュレート化（スクランブル解除）され、ユニット408で復号化され、ユニット410で誤り訂正され、ユニット412で暗号解除される。ユニット406、408、410及び412はそれぞれ図8の対応部310、308、306及び304の逆の機能を実行する。

【0093】

3. RMキャリアの生成

RMキャリアは、ふつうの人は見ることがほとんどできないように、検出されたり取り除かれたりしないように、また、送信メッセージを再生することが十分に確実にできるように、慎重に設計する必要がある。RMキャリア及びMH情報を格納するために使用することができるスペースは限られている。このことは、利用できる送信ビットの総数及びRMメッセージの再生性能に影響する。この他に、RMキャリアを含めたMPEGのビットの総数は、いくつかのレベルにおいてムービーポストハウス（movie post house）からの元のDVDビットストリームのビット数の総数に等しくなければならないという制約がある。そうでなければ、MPEG復号化におけるバッファオーバーフローやアンダーフローの問題が生じる可能性があり、ナビゲーション（navigation）及び元のDVDディスクの

他のDVD情報を変更しないでおくために、DVDの様々なセクタのビットストリームを編集することが必要となる。

【0094】

ビットの総数を変更しないでおくレベルはいろいろあり、これがキャリアの設計に影響を与える。最も簡単なレベルでは、1つ又は少数のビットを変更する。例えば、動きベクトルやDCT係数の符号ビットを変更したり、量子化スケールファクターを変更する。他のレベルでは、MBのビット総数を変更しないようにする。例えば、DCT係数を同じビット数のもので置き換えたり、ビット総数を変えずに複数のDCT係数を変更する。これらの2つの方法の主な有利な点は、一般にキャリアのサイズは小さいという点と、キャリアのための格納スペースが余分に必要ではない場合があるという点である。しかし、これらの方法には、大きくて制御することが難しい映像歪みがあり、メッセージ再生の可能性が低く、MHに適したMBの数が少ないという傾向がある。キャリアのビット数は、元のMBのビット数よりも大きい（又は小さい）ことが望ましい。キャリアの大きさが元のものより小さい場合と同様に、MHの全てのキャリアのサイズを同じにするために、ビット埋め込みが行われる。MPEGビットストリームの変更を確実に合法的にするため、MHのMBの後に新たなスライスヘッダが挿入される。同じ行のビット総数が変わらないように、その行の全てのMBを再符号化することによって、増加したビットの総数が補償される。再符号化の方法については、後により詳細に説明する。

【0095】

MBの周りの画素の強さを変えることによって、マクロブロックのエッジにおける変化を目に見えないように滑らかにする。例えば、図10のように、画素p(n)の周りの画素p(m)には、遷移領域を作るために、歪みが少し与えられる。

【0096】

本発明においては、RMキャリアを生成する基本的な方法は、映像源の強さを加えたり変えることである。この方法は、

- (1) 映像にノイズを与える

(2) エッジに沿って強度を変える

(3) テクスチャ領域の画像の特性を変える

というステップを含む。また、所定のメッセージホール (message hole) におけるビットの状態が変化することが1つのバイナリ値を示すようにし、ビットの状態が変化しなかったことが他のバイナリ値を示すようにしてもよい。動きベクトルの精度を1画素精度と半画素精度との間で切り換えてもよい。1つ以上の輝度ブロック又は色差ブロックのDCT係数を変更してもよい。更に、

(4) 所定の論理値のメッセージビットに応じて、MBの画像のエッジのプロファイルを修正する(図10bは、第1及び第2の論理値を示し、増加又は現象するエッジのプロファイルを図示している)。

(5) 図10cのように、画像のテクスチャを1画素、又は1画素以上若しくは1画素以下だけシフトし、所定の論理値のデータを示す。

(6) マクロブロックのテクスチャを伸長又は圧縮して示す(図10d)。

【0097】

このような方法や他の方法によってMBの変更し、メッセージのビットを定義することは、目に見えないように画像のテクスチャ部分において行くと有利である。同様に他の方法を採用することも可能である。重要な点は、変化して論理値を表すパラメータは、確実に検出されなければならない(ロバスト性がある)が、画像と混ざっていなければならないということである。

【0098】

ランニングマーク (running mark) の変調をするために映像にノイズを付加することは、歪みが小さく、コピーをする人に検出されにくいという点で有利である。図11に示すように、ノイズのようなキャリアは、同程度の歪みに対して、MB内で歪みを拡散する。このため、各画素の歪みは小さくなり、事実上目で見ることではできなくなる。カメラやフィルムからのノイズのように、ノイズは映像ソースに元から生じているので、ノイズのようなキャリアは、フレーム差分を求めても検出することは難しい。

【0099】

本発明では、MHの各DCTブロックの第1のnDCT AC係数を変更する

DCTによる方法がRMキャリアを生成するために用いられており、これは図12に示されている。配列MHBlock[0][block]が、MH(RMを除く)の元のDCTブロックを全て格納するために用いられる。配列MHBlock[1][block]及びMHBlock[2][block]は、リプレースメント(replacement)1及び2を作るためにノイズを加えた後のブロックを格納するために用いられる。キャリア0及び1は、元のブロック及び2つのリプレースメントから選択することができる。まず、nDCTの要素の配列がランダムに生成される。これらの要素は、0、1、及び2のいずれかの値を持つ。その割合は、例えば0が50%、1及び2はともに25%である。リプレースメント1を生成するために、要素が1及び2であれば、対応するDCT係数は固定値sigmaをそれぞれ加算及び減算する。リプレースメント1を生成する場合は、加算と減算とが逆になる。要素が0であれば、キャリアはともに元のDCTの値を用いる。nDCT及びsigmaの値は、輝度ブロックと色差ブロックとで異なるものとなりうる。図13は、この方法の例を示している。これは、nDCT=40、sigma=5、0、1、2の比が2:1:1である場合の輝度ブロックの画素領域における典型的なノイズパターンである。

【0100】

4. DVDプレーヤー環境におけるビット変調の実行

4.1 概要

DVDプレーヤー環境における図4aのビット変調器312の実行について、以下に詳細に説明する。

図4aにおけるBeは、総計Numbitビットであるとする。これらのビットは、ポインタBitsPtrで示される配列Bits[]に格納される。Bits[]の最後の配列要素が届くと、次に送るビットとして最初の要素が用いられる。映画には、RMビットの1つ以上のコピーが埋め込まれている点に注意する必要がある。コピーの数は、DVDディスク内の利用できるRMスペースによって異なる。

【0101】

MHの映像ストリーム、MHのサイズと位置、処理パラメータを置き換えるためのデータは、DVDのVOBU内のRMG_PCKと呼ばれるセクタに格納される。RMG_PCKのデータレイアウトとデータ構造は、図14、表3～5に

示されている。

【0102】

【表3】

DivxStreamId	アプリケーション
00000001b	ナビゲーションパック以外のDVDパックのメッセージホールビットストリームを置換することにより送信されたRMメッセージ。いかなるナビゲーションパックも走行マーク処理用の変更は不可。
その他	他の Divx アプリケーション用に予約

様々な Divx アプリケーション用プロバイダ定義ストリームの DivxStreamId

【0103】

【表4】

変数	説明
u32 bc	RMG ビットアレイ Bits[]内の RMG_PCK に送られる最初のビットを指定
u8 NumRMG	RMG_PCK 内の走行マーク回数
u8 NumBS	ビデオストリーム置換用に使用する付加ビットストリーム数。少なくとも1個の付加ビットストリーム(NumBS>0)を有し、(NumBS+1)の値が2のべき数であること。
u8 Reserved A	将来の修正用に予約
u1 FNextRMG	FNextRMG=0の時、全RMGデータ内にNextRMGデータと予約済み領域は存在しない。FNextRMG=1の時、全NextRMGデータが次のRMGデータの配置用に使用される。
u1 FSizeHole	RMG_PCK 内のメッセージホールのサイズが変わる場合、RMGデータ内にSizeHoleデータが存在し、FSizeHole=1である。それ以外は、FSizeHole=0であり、SizeHoleデータはRMGヘッダの後に発生する。
u8 ReplaceMode	メッセージホールのビットストリーム置換用に使用すべき適正なパラメータを有するアルゴリズムの決定

RMGヘッダのデータ構造

【0104】

【表5】

変数	説明
u16 NextRMG	RMG_PCK の始めから次の走行マークデータまでのバイトのオフセット。ゼロは連鎖の終わり。
u5 Reserved B	将来の修正用に予約
u11 SizeHole	走行マーク用メッセージホールのバイトサイズ
u32 SectorNum	メッセージホールが位置する V_PCK の論理セクタ数
u4 bcIncrMode	表6に記載の bc の増分を制御
u12 Offset	メッセージホールから SectorNum[]で指定された物理セクタの初めまでのバイトのオフセット。物理セクタ、バックヘッダ、パケットヘッダを含む。
u8 b[NumBS][SizeHole]	走行マーク 1 個の SizeHole 長さを有する付加 M P E G ビットストリーム
Reserved C	将来の修正用に予約

RMGデータのデータ構造

【0105】

図14aにおいて、RMG_PCKは、このRMG_PCKによって修正されるメッセージホールを含むV_PCKのどれよりも物理的に前に格納される。これらのV_PCKは、RMG_PCKが生じるVOBUの中にある必要はない。

図14bにおいて、RMG_PCKは、DVD private_stream_1 パックであって、そのパケットは供給者が定義したストリームによってエンコードされている。パケットのstream_id及びsub_stream_idは、それぞれ"1011 1101b"及び"1111 1111b"である。これは、DVDビデオ仕様書 (DVD video specification) 1.0のVi5-6ページの表5.1.1-1及び表5.1.1-2に記載されている。パックとパケットのヘッダは、DVDビデオ仕様書1.0のsection 5.2, Vi5-9に定義されている。DivxStreamIdは様々なDivxアプリケーションを示し、表3に定義されている。DivxStreamIdが"0000 0001b"である場合は、ランタイムモジュールは、本明細書に記載のアルゴリズム及びデータ構造に従って、ナビゲーションパックを除

いた全てのDVDパックのメッセージホールのビットストリームを置き換える。RMG_PCKのその他の部分は、RMGヘッダ、予約されたB, SizeHole及びRMGデータを有している。RMG_PCKの全てのデータは1つのセクタの中になければならない。すなわち、どのデータもセクタ境界を越えることはできない点に注意が必要である。

【0106】

図14c及び14dは、それぞれRMヘッダ及びRMデータの様々な変数を示している。これらの変数の目的は、表4及び5にまとめられている。RMヘッダはRMG_PCK内の全てのランニングマークによって用いられるRMアルゴリズム及びパラメータについてのグローバルな情報を提供する。NumRMGは、RMG_PCK内のランニングマークの総数である。各ランニングマークは、1つのRMGデータを用いる。変数ReplaceModeはメッセージホールのビットストリームの置き換えに使われる適切なパラメータとともに、アルゴリズムを決定する。

【0107】

例えば、図15のように、bcは、RMGビット配列Bit[]の中でRMG_PCKに最初に送られるビットを示している。RMが終了すると、bcの値は次のRMのために次の式（他の状況においては、他の関係が適用される）、

$$bc = bc + bcIncrValue \quad \dots (4.1)$$

に従って更新される。

【0108】

【表6】

bcIncrMode		bcIncrValue	公式
0	0000	1	bcIncrMode + 1
1	0001	2	
2	0010	3	
3	0011	4	
4	0100	5	
5	0101	6	
6	0110	7	
7	0111	8	
8	1000	0	8 - bcIncrMode
9	1001	-1	
10	1010	-2	
11	1011	-3	
12	1100	-4	
13	1101	-5	
14	1110	-6	
15	1111	-7	

bcIncrModeからbcIncr値を得るルックアップテーブル
bcIncr値は数式(1)におけるbcの更新増分値

【0109】

表6は、各RMのbcIncrModeから前記bcIncrValueを決定するためのルックアップテーブルである。新たなRMG_PCKのために、送信されるビットを再同期させるためにbcの値が用いられる。RMGビットはデータ型u16のBits[]に格納されているので、bcによって示され、送信されたビットを得るためには、適切なビット抽出が必要である。ここで、ランニングマークのために送られるべきビットの数を n_0 とする。これは、次の条件、

$$2n_0 \geq (\text{NumBS} + 1) \quad \dots (4.2)$$

を満たす最小の整数として定義される。

【0110】

NumBSはメッセージホールを置き換えるために用いられるRMデータ中のビッ

トストリームb[]の総数の増加分を示す。RMデータには少なくとも1つのビットストリームデータが加わっている必要があり、(NumBS+1)の値は2の累乗でなければならない。すなわち、NumBS > 0、かつ、n₀ > 0でなければならない。bclncrValueとn₀とは値が異なってもよい。V_PCKのメッセージホールの位置は、論理セクタ番号SectorNumとRMデータのu 1 2オフセットによって決定される。実行時のパフォーマンスをよくするために、RMG_PCK内のセクタ番号は全て減少しないように並んでいるものと仮定する。オフセットは、メッセージホールとSectorNumによって示された物理セクタ（物理セクタ、パック、パケットヘッダを含む）の開始点との間のバイト数を示している。RMG_PCK内のRMデータの追加ビットストリームの数は、RMデータのオーバーヘッドを減らすため、変更しない。メッセージホールのサイズはSizeHoleによって決定される。RMG_PCK内の全てのメッセージホールのサイズが変わらない場合は、RMGヘッダのFSizeHoleフラグは0にセットされ、RMGヘッダの後には1つのReserved B及び1つのSizeHoleのみが送られる。その他の場合は、各RMデータはReserved B及びSizeHoleを含む。この例では、1つのRMには1つのメッセージホールのみが処理される。

【0111】

実行時において、計算をすることなく、RMデータの予約領域をスキップしてRMG_PCK内の次のRMデータへ直接跳ぶために、NextRMGが使われる。予約領域は将来の発展のために使われる。NextRMGは、RMG_PCKの始めからRMデータの始めまでのオフセットのバイト数である。最後のRMデータについては、NextRMGには0がセットされる。RMヘッダのフラグFNextRMGが1にセットされると、各RMデータはNextRMGを有し、これらのNextRMGの値は、次のRMGデータへ行くために使われる。フラグFNextRMGが0にセットされると、各RMGデータには予約領域はなく、NextRMGが計算される。

【0112】

図16のアルゴリズムは、実行時におけるランニングマークの処理を概念的に示したものである。実際の実施形態は種々の再生ユニットの制約によって異なる。図16の3行目のRunTimeModule () プロセスは、DVDデータストリームの

最後に達するまで（又は、他の割込みが発生するまで）、実行中はVOBUのデータパックを読み続ける。8行目のreadSector () ルーチンは、VOBUから次のデータパックを返し、9行目のreadFromSectorルーチンは、このデータパックの特定の変数の値を返す。データパックのストリームタイプは、変数StreamIdによって決定される。データパックがビデオストリーム又はオーディオストリームのいずれか（すなわち、V_PCK、A_PCK又は他のオーディオパック）であれば、復号化され、その結果は表示のためにビデオ又はオーディオポートに送られる。他の型のパックについては、対応する処理が進められる。ランニングマークプロセスは、RMG_PCKが読み込まれ、すなわち、このデータパックは供給者が定義した（private stream 1の）ストリームであり、そのDivxStreamIdは"0000 0001b"に等しい。その後、RMGヘッダが26行目のように読み込まれる。メッセージホールのサイズが変わっていなければ、すなわち、FSizeHole = 0であれば、28行目のようにSizeHoleが読み込まれる。このRMG_PCKの全てのランニングマークは、12行目のRunningMarking () ルーチンを各ビデオセクタ毎に呼び出すことによって行われる。これらの処理においてエラーが生じた場合には、適切なエラー処理動作が開始される。

【0113】

48行目のRunningMarking () ルーチンは、現在のビデオパックを参照するRMデータを適用することによってランニングマークを行う。次のRMデータの位置のNextRMGは、RMG_PCKから読み出され、又は計算によって求められる。これは、フラグFNextRMGの値による。FSizeHoleが1にセットされている場合は、SizeHoleフィールドが読み出される。その後、セクタ番号、ビットカウントインクリメントモード（bit count increment mode）、メッセージホールのオフセット、及び付加されたビットストリームb[]が読み出される。セクタ番号が現在のビデオセクタの番号に一致するときは、66行目のReplaceMessageHoleData () が呼び出され、ReplaceModeに従ってMHビットを置き換える。次のRMG_DATA構造に移る前に、b cはbclncrModeを使って更新される。

【0114】

74行目のReplaceMessageHoleData () は、変数のモードに従って種々の方法

でMHビットの置き換えを行う。Mode=0の場合は、MHビットはb[0]で置き換えられる。Modeが1又は2の場合は、図15のように、bcによって示されたBits[]からn₀ビットを抽出することによって、sendBitが作られる。Mode=1の場合、MHビットはb[sendbit-1]で置き換えられる。version=2の場合は、sendbitの下位n₀ビットが反転される。すなわち、n₀=3の場合、“0000 0101”が“00 00 0010”になる。その後、MHビットはb[sendbit-1]に置き換えられる。

【0115】

ランタイムモジュールのファームウェアのクラッシュを防ぎ、RMG_PCKのエラーに起因する映像の歪みを最小限にするため、実施の際には適切なチェックが必要である。誤りがあると、MHビットの置き換えが行われず、処理は次のランニングマークに移ってしまう。

【0116】

4. 2 ランニングマーク位置ホルダーとランニングマークパック

ランニングマークパック（ここでは“Divx Packと称する）は、パケットが、供給者が定義したストリームとして符号化された、DVD private_stream_1パックである。パケットのstream_id及びsub_stream_idは、それぞれ“1011 1101b”，“1111 1111b”に等しく、これらはDVDビデオ仕様書1.0のVI5-6ページの表5.1.1-1及び表5.1.1-2に記載されている。sub_stream_idの直後のサブピクチャデータ領域には、“0x 4449 5658 312e 3023”（16進数）に等しい64ビットがある。これらのビットは、文字列“DIVX1.0#”を表している。パケットの残りのビットは0にセットされている。

【0117】

Divxパックは、Divxパック周波数において、DVDビットストリームに様に分布している。これは次の式、

$$N_{divx} = \text{truncate}((\text{MovieLength}/(15*\text{factor}))+0.5)$$

に従って決定される。

但し、N_{divx}はD_{inx}パック間にはさまれるMPEG符号化ビデオフレームの数である。MovieLengthは、タイトル作成者によって特定された分単位での再生長さである。NTSCの係数は1.0、PALの係数は1.2で

ある。Ndivxの計算値が2未満の場合、Ndivxは2に設定する。Divxパックは物理的に（一時的にではなく）ビデオパックに対して位置している。Ndivx符号化ビデオフレームの収集はDivxビデオグループとして引用する。Divxパックは、Divxビデオグループの第1バイトを含む付随のビデオパックの前に物理的に起こる。その結果、DivxパックはDVDの第1ビデオパックの前に現れる。Divxパックの表示は一時的に付随ビデオパックと無関係となる。Divxパックの表示時間は、付随Divxビデオグループ前、又は後に一時的に起こる。

【0118】

Ndivxの計算を示すため、85分間のNTSC映画を仮定する。公式から、Ndivxは6である。したがって、Divxパックが6符号化ビデオフレームごとにDVDビットストリームに挿入され、Divxビデオグループが6つのビデオフレームを含むことになる。各符号化ビデオフレームが1.2のビデオパックを取り、Divxビデオグループが7.2のビデオパックからなると仮定する。Divxパックのないビデオグループのレイアウトを図17に示す。Divxビデオグループ1からの最後のビデオフレームの部分が、ビデオグループ2からの第1ビデオフレームと同じパックに存在している。Divxパック挿入後のビデオグループとDivxパックのレイアウトを図18に示す。グループ2のDivxパックは、同一パックのグループ1のビデオが存在していても、グループ2の符号化ビデオの第1バイトを含むパックに先行している。

【0119】

Divxパックで創られたDVDディスク画像はMEI照合を通る必要がある。Divxパックは有効個人__ストリーム__1パックである。これは、パックが有効SCRを含み、DivxパケットがDVDビデオ仕様書のV15-29、及び30において定められた必要な値を含んでいることを意味する。PES__Scramble__Controlフィールドは“スクランブルなし”に設定する。Original__Or__Copyフィールドは、“オリジナル”に設定する。各Divxパックは有効PTSを含む必要がある。このPTS値はビデオのPTS値と関係している必要はない。仕様書のV15-30頁のNote2に必要な

VOBの第1パックを除いては、PTSデータが続くPES個人データはない。

【0120】

DIVxパックがその付随ビデオパックに物理的に隣接している必要はない。ナビゲーション、オーディオ、又は他のサブ画像パックはDIVxパックとその付随ビデオパックの間で起こる。DIVxビデオグループがBlock、PGC、VOB、セル、またはVOBUを完全に含んでいる必要はない。DIVxビデオグループは、各境界に広がることができる。DIVxビデオグループは新たなBOVの始まりで再スタートすることができる。

【0121】

タイトル作成者は、DIVxパックの作成を可能、不可能の間で選択できる。この作成はデフォルトで不可能となる。DIVxパックの作成が可能な場合、タイトル作成者は、Ndix計算のために映画の分単位で時間を特定できる。タイトル作成者が書き入れた映画の時間が15分未満、又は240分以上の場合、確認メッセージが現れ、タイトル作成者は書き入れた値を受け入れたり、拒絶したりできる。タイトル作成者は、書き入れた映画時間から決定する計算したNdixを上書きすることができる。MovieLengthの値はNdixの新しい値では更新されない。Ndixの値は2未満、又は2030-1以上としては特定できない。Ndixの上書き値が有効でなく、変更が失敗の場合、エラーメッセージが現れる。DIVxパックの作成ができる場合は、タイトル作成者は映画時間、又はNdixのいずれかを特定しなければならない。

【0122】

5. メッセージホールの生成及び再配置

RMシステムにおいて、ビデオを構成するMBのいくつかがMHとされる。MHとなるMBの選択方法が必要である。この選択はRMシステムにとって決定的ではない。情報（メッセージ）は、MHとしてのMBのコンフィギュレーションを仮想的に用いることで表現されかつ再生され得る。したがって、MHの選択のための可能性のある多数の技術を採用し得る。

【0123】

好ましい実施形態において、MHの選択は2つのフェーズからなる。すなわち

、1) MH候補の初期選択(メッセージホールの生成)に続き、2) これらの候補からの最終選択(メッセージホールの再配置)である。この2フェーズ手法は、いくつかの理由のうち、各Bフレームに程良い数のMHを有することが望ましいとの理由から採用される。メッセージホール生成(第1フェーズ)では、本来必要な数より多くの候補が選択される。再配置(第2フェーズ)では、1つのBフレームあたりの最終的なMHの数が相当に程良い数になるように、Bフレーム中の過剰ないくつかの候補が捨てられる。以下の節に述べるように、その他の種々の理由から候補を捨ててもよい。

【0124】

RMシステムの本形態では、ムービー中に配置できるMHの総数は、サブピクチャチャンネル中の空き領域と、MHの大きさと、DVDが再生される際のビデオにリアルタイムにRMを挿入するためのDVDプレーヤの計算上の容量とにより制限されている。1つのBフレームあたりのMHの平均数は、一般に12より少ない。MHの配置にあたって考慮すべき点は、(1) MHの位置が予測不能であること、(2) MHが複数フレームに渡ってできるだけ均一に配置されること、(3) RMがMHの中に挿入された際にMHが視認されにくいことである。MHの位置が予測不能であるべきなのは、例えばMHの位置が既知であり又は予測されると、ムービー海賊版製作者が伝達情報(メッセージ)をたやすく改竄することができるからである。MHが複数フレームに渡ってできるだけ均一に配置されるべきであるのは、リアルタイムにMH中にRMを挿入するのに必要な計算上の容量が一定でないからである。加えて、単一フレーム又は少数のフレームに多数のMHが配置されると、これらのフレームを削除することにより、かなりの量の情報が消失又は除去されてしまうからである。RM挿入の後にビデオ視聴者にとってMHが見えにくい(又は少なくともあまり気付かれない)ことは、ビデオの品質がRMシステムによってあまり悪影響を受けないことに通じる。また、MHが見えやすいと、改竄が容易になってしまう。

【0125】

1つのアプローチは、人手で又は半自動で各フレーム中の所要数のMHを選択することである。例えば、ムービーを構成する各Bフレームをコンピュータプロ

グラムが表示し、熟練したオペレータがプログラムを使ってMHとして使われるべきMBを選択するのである。このアプローチは、MHの選択に人間の目と人間の洞察力を用いる点で優れている。例えば、気付かれにくいようにMHが配置されるからである。このようなアプローチは、処理すべきフレームが多数にのぼるので、出費がかさみ、時間がかかり、かつ退屈である。例えば、110分のムービーには158400フレームがあるのである。したがって、自動MH生成が重要になってくる。本発明に係るアプローチによれば、乱数ジェネレータを用いることでHMがランダムに生成される。MHの視認性を低減するための他のアプローチもある。最初に1フレームの全てのMBを平滑と非平滑（テクスチャあり）とに分類し、それから「テクスチャあり」の集合の中からMHとしてMBをランダムに選択するのである。これら2つのアプローチは、5.1節及び5.2節においてそれぞれ述べる。

【0126】

5.1 MHのランダム生成

MH生成に要求されるのは、その位置が予測可能なパターンに従わないようにすることである。この要求は、MHをランダムに配置することによって満たされる。例えば、疑似乱数ジェネレータを用いてMH配置を取り上げるのである。この方法によれば、1フレームを構成するMBの各ロウに固定数の候補MHがランダムに配置される。例えば、仮に1ロウあたりの候補MHの所要数が3であるものとする、アルゴリズムにより各MBロウに3個のMHが配置されるが、MHの水平位置は、均一分布を持つ疑似乱数ジェネレータを用いてランダムに割り当てられる。各MBロウについては、既に候補MHとして選択されたMBのリストをアルゴリズムが持つことで、同じMBが重ねて選択されず、かつ全てのロウが候補MHの目標数（例えば3）を持つようになされる。例えば、疑似乱数ジェネレータがたまたま同じMBを2度目に選択したなら、このMBが候補MHのリストの中にあることが見い出され、リスト中に存在しないMBが見つかるまで疑似ランダム選択のプロセスが繰り返される。一旦1フレーム中の全てのMHが選択されると、これらがMBアドレス（MBA）の昇順にソートされる。昇順ソートの理由は、RMシステム中の他のアルゴリズムがMHをこの順に処理することに

なっているからである。

【0127】

このアルゴリズムの疑似コード表現が、図19に示されている。このアルゴリズムのパラメータは、`Width`、`Height`、`nFrames`、`numMHrow`及び`seed`である。`Width`及び`Height`の値は、フレームサイズをピクセル単位で指定する。1セットのフレーム中の各MBロウのために生成された候補MHの数が`numMHrow`である。`nFrames`パラメータは、処理されるべきフレームの数を指定する。候補MHのMBアドレスリスト、すなわち`MHAlist`が本アルゴリズムの出力である。

【0128】

図19の`gMHrow`のアルゴリズムによれば、`Width`、`Height`、`nFrames`、`numMHrow`及び`seed`の値が得られる。そして、MBロウ中のMBの数、すなわち`MB_Width`が計算され、`numMHrow`が0から`MB_Width`までの範囲に制限される。ここに、`MB_Width`は1MBロウ中のMBの数である。疑似乱数ジェネレータは`seed`パラメータにより指定された「種」の値によって初期化される。疑似乱数ジェネレータを用いて各フレームの各ロウにMHが生成され、`lowMBA`と`highMBA`との間の乱整数に戻る。本アルゴリズムによれば、1フレーム内でMHの繰り返しが無いことが保証される。そして、`MHAlist`の中の全てのMBAが昇順にソートされて出力される。

【0129】

5. 2 領域分類を伴うメッセージホール生成

更に洗練された他のMH選択のやり方は、人がビデオの歪を許容し得るビデオ中の領域を画定し、これらの領域の中にMHを配置することである。このやり方は、空間的、時間的、輝度及び／又は色の特性に従って、ピクチャフレームを異なる領域に分割するものである。ここでは、MBが平滑と非平滑（テクスチャあり）とに分類される。「テクスチャあり」と分類されたMBが候補MHとしてランダムに選択される。これら2つのプロセスについては、次の段落において説明する。人間の目は非平滑領域中に比べて平滑領域中の歪みに対して敏感なので、

平滑領域へはMHを配置しない。

【0130】

Chun et al., "An Adaptive Perceptual Quantization Algorithm for Video coding", IEEE Transactions Consumer Electronics, 39(3): 555-8, Aug. 1993
と同様のアプローチにより、MBのルミナンスブロックのDCT係数を用いて、2ステップで分類が実行される。この分類アルゴリズムによれば、ピクチャフレーム中のMBがランダムに選択される。取り上げられたMBが次の段落で説明するいくつかの基準を満たすならば、当該MBが平滑又は非平滑（テクスチャあり）に分類される。MBが「テクスチャあり」に分類されると、当該MBが候補MHとみなされる。このプロセスは、所要数のMBが選択されるまで、又はフレーム中に「テクスチャあり」のブロックが残らなくなる（すなわち、全ての「テクスチャあり」のMBが候補MHとして受け入れられる）まで繰り返される。

【0131】

分類の前にMBが満たさねばならない基準の1つは、それがフレームの境界に位置しないことである。ここで境界とは、MBの最初のロウから最後のロウまで、及びMBの最初のカラムから最後のカラムまでからなる、MBの外側の矩形のことである。これらのMBを外して考える理由は2つある。第1の理由は、家庭用VCRのような安価なアナログ記録装置では、これらのMBがしばしば特に低い忠実度で再生されるからである。したがって、それらの装置で記録された場合には、これらの領域からRM情報を引き出すことは困難である。第2の理由は、多くのテレビジョン受像機ではこれらの領域が表示されず、ピクチャフレームの表示領域から外にはみ出してしまうからである。これらのMBは、例えば黒領域に置き換えられる。この置き換えにより、ビデオの目視性に悪影響を及ぼさずに、これらの領域中のRM情報を効率的に除去することができる。

【0132】

MBが満たさねばならない他の基準は、大多数の画素ルミナンス（グレースケール）の値が、受け入れ得る範囲の値であるということである。MBをデコード（伸張）すれば256（ 16×16 ）個の画素値が得られる。例えば、25%より少なければ画素ルミナンス値は25から230までの範囲外であってよいと定

めることができる。この場合には、画素値のうちの64個以上が25より小さいか又は230より大きいならば、当該MBは分類の対象とならず、したがって候補MBとはなり得ない。画素値のパーセンテージと、受け入れ得る画素ルミナンス値の範囲とは、アルゴリズムのパラメータである。ただし、必要に応じて調整されるものであり、例示の値(25%、25、230)に限定されない。本例で採用した特定の値は、実験及び技術者の判断により決定されたものである。この基準のきっかけは、明るすぎたり暗すぎたりする画素が、しばしばMPEG再エンコーディング(後述)により、又は例えば家庭用VCRによるアナログ記録によりクリップ(それぞれ増加又は減少)されることにある。ただし、画素値の範囲が減縮されるので、このクリッピング効果がRM情報の再生又は抽出の能力に悪影響を及ぼすことがある。

【0133】

MBが満たさねばならない更に他の基準は、前フレーム中の同じ位置のMBがMHとなるように選択されていないことである。この基準により、MHは隣接フレーム中の同じ位置に配置されない。同じ位置に出現するRMは、視認されやすい(気付かれやすい)。

【0134】

平滑か非平滑(テクスチャあり)かというMBの分類は、まずMBを構成する4個のルミナンスブロックを分類することから始まる。これら4個の分類結果は、MBの分類を決定付けるように、評価されかつ結合される。ここで、あるDCTブロックの水平、垂直及び対角線方向のエネルギーをそれぞれ E_h 、 E_v 及び E_d で表すこととする。 E_h の値は、次のテンプレート中の「h」で示されたDCT係数の2乗和により計算される。

【0135】

```
. d h h h h . .
d d h h h h . .
v v d d . . . .
v v d d . . . .
v v . . . . .
```


.....

同様に、 E_v 及び E_d の値は、それぞれ「v」及び「d」のマークが付けられた要素の2乗和により計算される。このテンプレートの各位置は、ルミナンスブロックを構成するDCT係数の 8×8 行列の要素を表している。このテンプレート中の左上のDCT係数は、DC項である。本ブロックの水平、垂直及び対角線方向のエネルギーの平均を E_a とする。これは、 E_h 、 E_d 及び E_v の平均である。そのうえで、ブロックが次の規則に従って分類される。

【0136】

もし($E_a < T_1$)ならば、

当該ブロックは平滑ブロックである：

さもなければ、

当該ブロックは「テクスチャあり」のブロックである。

【0137】

E_a は、ブロックが平滑か非平滑かの分類に用いられる。しきい値 T_1 は、一連の実験により、及び技術者の判断により決定し得る。その値は固定である。しかしながら、このしきい値(T_1)は、ビデオ又はその一部の特性に基づいて自動的に決定することも可能である。そして、その値は時間の経過とともに変動してもよい。例えば、1フレーム毎に異なる T_1 の値を用いる。

【0138】

MBを構成する4個のルミナンスブロックの分類から、次の規則に従って当該MBが分類される。

【0139】

もし(当該MB中の3又は4ブロックが「テクスチャあり」)ならば、

当該MBは「テクスチャあり」のMBである：

さもなければ、

当該MBは平滑MBである。

【0140】

上記分類ルーチンの疑似コードを図20に示す。この分類ルーチンによってコールされる初期化関数の疑似コードを図21に示す。はじめに、全てのMBが分類に利用し得るものと仮定する。そして、前Bフレームで選択されたMHの位置が、現フレームでは利用不能としてマークされる。これにより、隣接Bフレームの同じ位置にMHが出現することを防止できる。MH選択の他の制約は、MHがフレームの境界にあってはならないということである。これらのMBも利用不能としてマークされる。分類ルーチンは、フレーム中の予め特定された部分のみを処理する。つまり、調べられるMBの数は最大限が規定されている。サーチされるべきBフレームのパーセンテージを制限することにより、候補MHとして取り上げられるMHのパーセンテージが低減される。したがって、隣接フレームの同じ位置に2個のMHが出現する確率が低減される。なぜなら、サーチされる領域はランダムに決定され、かつフレーム毎に異なるという傾向があるからである。所要数のMHを見い出せなかった場合には、アルゴリズムが終了する。

【0141】

初期化ルーチンがコールされた後、与えられたBフレーム中のMHのサーチが開始する。アルゴリズムは、フレーム中のMHがなくなるか、又は所要数のMHが見い出された場合に終了する。MBのロウ及びカラムは、ランダムに選択されたMBAから計算される。MBは、たいていのルミナンス画素値が規定範囲内でないなら、選択されることはない。そして、MBが分類される。MB分類関数（`Classify_MacroBlock`）の疑似コードを次に示す。MBが「テクスチャあり」に分類された場合には、当該MBが取り上げられるようにマークされる。ロウ上のMHの数が1ロウあたりの最大許容数に等しいならば、当該ロウからはもはやMHを取り上げない。この制約は、再エンコーディングにとって必要である。MBが平滑であると分類された場合には、これが用いられることはない。

【0142】

図22にはMB分類の疑似コードが示されている。8×8ブロックのDCTが計算された後、DCT行列中の高エネルギー成分が水平、垂直及び対角線方向のエネルギー E_v 、 E_h 及び E_d として計算される。これら3値の平均、すなわち E_a

が計算される。「テクスチャあり」との分類の 8×8 ブロックの数が数えられる。もし3個又は4個のブロックが「テクスチャあり」ならば当該MBは「テクスチャあり」に分類され、さもなければ分類は平滑となる。

【0143】

5.3 メッセージホールの再配置

MH選択の第2の(かつ最終の)フェーズは、再配置である。再配置の目的は、フレーム間でMHをできるだけ均一に配置することにある。いくつかのフレームの中にMHの塊ができることがないように、各フレーム中に程良い数のMHを配置することが好ましい。この配置は、DVDプレーヤによるリアルタイムRM挿入の間のほぼ一定の計算上の負担となる。また、いくつかのフレーム中に多数のMHが配置されると、これらのフレームの削除によって、かなりの量の情報が除去される結果となる。再配置では、過剰の候補を有するBフレームからいくつかの候補MHを捨てることにより、1つのBフレームあたりのMHの数がほぼ一定に保たれる。

【0144】

再配置アルゴリズムは、ビデオを隣接グループ又は隣接セグメントの単位で処理する。これをセクションと呼んでいる。ビデオのグルーピングに対応したセクションは、そのままDVDの中に格納される。セクションは、ビデオのほとんど任意のポイントで開始及び終了することができる。この状況が図23に示されている。この例では、セクションがあるフレーム中のある特定のMB(左側の小さい黒正方形)から始まっている。そして、当該フレームの残りに続き、いくつかのフレーム全部と、他のフレームの一部(グレー表示)とを経て、最終フレーム中のあるMB(右側の小さい黒正方形)で終わっている。セクションは、任意のタイプのフレーム(I、P又はB)で開始したり終了したりすることができる。

【0145】

本例では、DVD上の制限された容量の補助記憶が、RMシステムに割り当てられている。この制限は、1セクションあたり2Kバイトである。この制限のために、最小の補助記憶容量を占めるMHの方が、大きい補助記憶容量を占めるMHより好ましい。この意味から、MHのサイズは、MHに関連する補助記憶のサ

イズ（ビット単位）であると考えられる。再配置は、大きいサイズのMHより前に小さいサイズのMHを処理する傾向がある。

【0146】

再配置の基本コンセプトは複雑なものではない。セクション中の各Bフレームに対して、そのフレームについては最終の（捨てられない）MHとなるように、1組の候補MHの中から最小のMHを選択する。候補MHからこれら最終MHを、再び選択されることがないように除去する。補助記憶の容量（例えば2Kバイト）がなくなるまで、又は候補MHがなくなるまで、この処理を続ける。補助記憶がなくなってしまう場合には、残りの候補MHは捨てられる（用いられない）。この処理は、少なくともあるセクションにおいて、1つのBフレームあたりの程良い数の最終MHを与える傾向がある。もちろん、あるBフレームについて候補MHが少ししかない場合には、例えば当該フレームが基本的に平滑MBからなる場合がそうであって、Bフレームが候補MHを全く持たないこともあり得るので、このようなケースでは最終MHを少ししか又は全く持たない。

【0147】

実用上は、上記基本コンセプトにいくつかの改良が必要である。1つの改良は、補助記憶がなくなった場合のMHの配置に関するものである。ここで、セクションに5つのBフレーム（「a」から「e」まで）があるものとし、第3番目のBフレーム（「c」）の後に補助記憶がなくなったものとする。各Bフレーム中のMHの数は、

Bフレーム	a	b	c	d	e
MHの数	7	7	7	6	6

のようになる。

【0148】

最初の3個のBフレームの中には7個のMHがあるが、最後の2個のBフレームの中には6個のMHしかない。好ましい配置は、

Bフレーム	a	b	c	d	e
MHの数	7	6	7	6	7

である。

【0149】

(「c」の後に) 補助記憶がなくなったとき、Bフレーム「a」が最後に処理された状態までアルゴリズムが戻る。この時点では、各Bフレームに6個のMHが割り当てられる。候補MHは、好ましい配置が得られるように、フレーム「a」、「c」及び「e」の各々から選択される。あるフレームが利用可能な候補MHをもはや持たないならば、利用可能なMHが発見されるまで両方向に動くこととなる。

【0150】

基本コンセプトに対するもう一つの改良は、セクションの始まりと終わりに発生し得る部分フレームに関するものである(図23参照)。セクションの最初もしくは最後のフレームがBフレームの場合、部分Bフレームを生成する必要があるかも知れない。この場合、部分Bフレーム用に選択されるMH数はセクション内に存在するBフレームの比率に従って割り当てられる。例えば、Bフレーム領域の40%がセクション内にある場合、部分フレーム用に選択されるMH数は全フレーム用に選択される数の40%とすべきである。

【0151】

アルゴリズムの擬似コードを図24に示す。全ての変数をinitialize()ファンクションで初期化する。initialize()ファンクションを図25に示す。セクションの始まりがIフレームもしくはPフレームにある場合、IフレームまたはPフレームに続く最初のBフレームを最初のフレームとして選択し、そのフレームの最初のMBをセクションの開始MBとして採用する。同様に、セクションがIフレームまたはPフレームで終わっていれば、そのセクションの最後のBフレームがセクションの最後のフレームになる。

【0152】

セクションがBフレームの中央にある場合、フレームの一部を前のセクションの前の呼出しで処理する。次のセクションのために、この情報を格納して前のセクションで選択されたMHが再度選択されないようにする必要がある。DVD境界線に沿うMHは、再配置に使用されない。

【0153】

初期化の後に（図24参照）、再配置が始まる。最初のwhileループは、利用可能なMHがなくなるか、又は選択されたMHのサイズが1セクションあたりの最大サイズ（例えば2Kバイト）を超えた場合に終了する。各フレームから探し当てられた最小サイズのMHは、最終MHとして選択される。

【0154】

最初の再配置がなされた後、最後の繰り返し中で選択されたMHが再度の再配置にかけられる。最後の繰り返し中で選択されたMHは除去される。更に均一な配置の中から他のMHが選択される。全てのMHが選択された後、これらがアルゴリズムから出力される。

【0155】

6. MPEG再符号化

MPEG再符号化は、RMの挿入後、一以上のMHを備えた一列内のMBのビット総数を、RM挿入前の総数と等しく維持することを目的とする。MHの余分なビット数は、より少数のビットとすることで、その列内のMBに充填される。原符号化と再符号化のビット数をそれぞれその列用にOrigNumBitsとRowBitCntと定義し、その差を次のようなBitsDiffと定義する。

【0156】

$$\text{BitsDiff} = \text{OrigNumBits} - \text{RowBitCnt} \quad (6.1)$$

本発明に対応する符号化のためにより少ないビット数を用いるMBを作成するアプローチは、要素mquantを尺度化しながら量子化を減少することである。一般に、MBのMquantをより高くすればするほど、ビット数の利用はより減少するが、より大きな歪みが生じる。このため、一般には、MBのMquant値の変化が正確にOrigNumBits以下かそれと同等の（つまりBitsDiff<=0）Rowbitsを作成する。ビットの詰め込みは両者をその後等しくするために利用される。計算の複雑さは増すが、MBの幾つかのDCT係数の歪みをより調整することにより、ビットを詰め込むためのわずかなビットが得られる。

【0157】

各MBのMquantの増加量は分からないので、反復検索は再符号化が必要となる。この反復検索は、BitsDiffがゼロ以下の状態ならば、終了する。そして

繰り返し数は $iterMax$ と定義された許可数に到達するか、もとの $mquant$ 値の増加は最大許容歪み値 $MquantIncMax$ を越える。後者の二つのケースでは、その列内の全ての MH は除かれ、RM メッセージビットを送信するために、その列内の RB は全く使われない。MB 内で歪みを統一的に拡散するために、MB の $mquant$ 値は同等の量だけ増加される。再復号化の過程をスピードアップするために、全ての必要な符号化情報は、一以上の MH を含む列群のための MPEG ビット列を復号することで抽出される。この情報は動きベクトル、符号化ブロックパターン、原 $mquant$ 値、動きと DCT タイプ、MB が飛ばされているかいないかの決定を含む。もし列内に MH が全く存在しないなら、その列の再符号化過程は飛ばされ、原 MPEG ビット列が使われる。

【0158】

〔表 7〕

iterStatus	説明
Inc_delta_Row	再符号化後のビット数が元のビット数にまだ程遠い。BitsDiff \leq 0 の条件を最初に満たす前の反復状態として定義される。
Dec_delta_Row	再符号化後のビット数が元のビット数にまだ程遠い。BitsDiff \geq 0 の条件を最初に満たす前の反復状態として定義される。
Inc_Mquant_Row	再符号化後のビット数をできるだけ元のビット数に近づけるために状態 Dec_delta_Row に到達後に設定される反復状態。
Dec_Mquant_Row	再符号化後のビット数をできるだけ元のビット数に近づけるために状態 Inc_delta_Row に到達後に設定される反復状態。
Inc_Mquant_MB	再符号化後のビット数が元のビット数に近い。BitsDiff \leq 0 の条件を二度目に満たす前の反復状態として定義される。
Final_iteration	BitsDiff=0 の条件を満たした後の反復状態として定義される。 mquant 値の変更は不可。もう一度再符号化して MH のビデオストリームデータとビットストリームデータを送出する。その後、反復ループを出る。

再符号化の反復状態

【0159】

各繰り返しを読み込まれている間、再符号化状態の決定を助けるために、そのプログラムは、表7で定義された多様な `iterStatus` を使用し最新化する。`iterStatus` である `Inc_delt_Row` と `Inc_Mquant_Row` は、各選択されたデルタか1により列内の全てのMBの `mquant` 値を増加させるために、そのプログラムを指示するよう使われる。同様に、`iterStatus` である `Dec_delt_Row` と `Dec_Mquant_Row` は、各選択されたデルタか1により列内の全てのMBの `mquant` 値を減少させるためにそのプログラムを指示するよう使われる。

【0160】

6.1 MPEG再符号化技術

本発明における第1の実施形態の再符号化によると、一のMBが選択され、その `mquant` 値は一つだけ増加し、全体の列は再符号化される。もし上記した最終条件が満たされるなら、繰り返し過程は止まる。さもないと、もう一つのMBが選択され繰り返し過程は継続する。`mquant` 値は各繰り返しに対して維持され、前の繰り返しに残ることはないであろうことに注意しなさい。MPEG符号化にとり、前のMBの `mquant` 値と異なるだけの場合、一のMBのそれは符号化され送信されるので、ある列の選ばれたMBは連続して使われる。その列の最後のMBが届いた時、最初のMBは次の選択のために再び選ばれるだろう。図26はこの繰り返し過程の典型的な結果を示す。最初の $(j+1)$ のMBの `mquant` 値が $(d+1)$ の値だけ増加され、MBの最後のその値が d の値だけ増加され、そこでは、`MB_width` はある列内のMBの総数となる。

【0161】

このアプローチのアルゴリズムが図27に表わしてある。6行目のように、現フレームの各MH列にとり、(一回で一つの)列内で選択されたMBの `mquant` 値は引き続き1だけ増加される。各再符号化が行なわれた後、`BitsDiff` の条件がゼロ以下を満足してないなら、その列の次のMBが21行目であるように選択される。このアプローチの利点は、実行は容易だが、多数の繰り返しが必要となる。

【0162】

第2に実施形態では、表7における多様な `iterStatus` は `Inc_Mquant_Row` となるように最初にセットされる。各繰り返しのために、その列内の全てのMBの `mquant` 値はデルタにより増加され、全ての列は再符号化される。デルタが1であると仮定しよう。このプロセスは `BitsDiff` がゼロ以下になるまで続く。そして、`iterStatus` は `Inc_Mquant_MB` にセットされ、その `mquant` 値は、そのデルタを減じるにより前の繰り返しの `Inc_Mquant_MB` に再セットされる。次に、その列の最初のMBから始め、第1のアプローチは `BitsDiff` がゼロ以下になるまで使われる。`iterStatus` は、その後、`Final_iteration` と等しくセットされ、最後の繰り返しは、映像とMBのビット列を送信するための繰り返しループを通して行なわれる。

【0163】

本発明の側面である第2の実施形態を説明するアルゴリズムが図28に与えられている。多様な `iterStatus` は、3行目にあるように `Inc_Mquant_Row` と等しく最初にセットされる。各読み込みに対して、列内の全てのMBの `mquant` 値は、その列を符号化するのに必要なビット数が標的となるビットレート以下になるまで、7行目にあるようにステップサイズのデルタだけその後増加される。しかしながら、再符号化された列のサイズを可能な限り元の列のサイズに近づけるために、その列内のMBの全ての `mquant` 値は、25行目にあるようにデルタにより減少され、そして、`iterStatus` は26行目にあるように `Inc_Mquant_MB` と等しくセットされる。次に、27行目にあるようにその列内の最初のMBから始め、各MBの `mquant` 値は、その列を符号化するのに必要なビット数が標的となるビットレート以下となるまで、最初の実施形態でのように一度につき一つ変化される。この点で、`iterStatus` は、`Final_iteration` と等しくセットされ、最後の読み込みが映像とMBのビット列データを送信するための繰り返しループを通して行なわれる。

【0164】

第2の実施形態はより多くのプロセスが必要となるが、第1のアプローチと比べた繰り返しの節約はかなりである。`NumIter(i)` を i 番目のアプローチのための繰り返し数と定義する。第1の実施形態では、次の式で表わされる。

【0165】

$$\text{Numlter}(1)=1+d \times \text{MB_width}+(j+1)+1$$

(6.2)

$$\text{Numlter}(2)=1+(d+1)+(j+1)+1$$

(6.3)

両実施形態においては、最初の繰返しは、MBの原 mquant 値を用いた最初の読み込みのためであり、最後の繰返しは、映像とMBのビット列データを送信するための最後の読み込みのためである。このため、第1のアプローチを越える第2のアプローチを用いた繰返しの節約は、次の式

$$\text{Numlter}(1)-\text{Numlter}(2)=d \times (\text{MB_width}-1)-1 \quad (6.4)$$

で与えられる。

【0166】

720×480画素のフレームにとり、一列に45MBが必要となる。典型的なMPEGビデオでは、 $d=4$ である。この場合、175回の繰返しを節約できる。

【0167】

本発明によるMPEG再符号化のための第3の実施形態は、第2の実施形態を越え洗練されたものであり、そこでは図26における j 値が繰返しの前後の間で列内の全てのMBのビット数の相違を比較することで計算される。その後、列内の最初のMBから始める代わりに（図26のように）列内のMBの mquant 値を用いることにより、その再符号化は第1の実施形態と切り替わる。 j 値の概算が一般に現実の値と非常に近いので、 BisDiff の状態をゼロ以下に到達させるために繰返し数を増やす数は、一回か二回と非常に少ない。二番目のアプローチを越える三番目のアプローチの繰返し数の節約は j 値に基づくことになる。 j は0と $(\text{MB_width}-1)$ の間のいかなる値をも取りうるので、平均的な節約数は、 $\text{MB_width}/2$ となる。再び720×480画素のフレームにとっては、一列内に45MBが存在する。それは、約22の繰返しが節約されることを意味する。

B_i を k 番目の繰返しの i 番目の符号化のビット数と定義する。全てのMB上

の $B_i(k)$ の合計は k 番目の繰り返しの $RowBitCnt$ と等しい。図 26 における $quant$ の値を増加するために、 $(d+2)$ の繰り返しの後、 $BisDiff$ がゼロ以下の状態になることを、その図は示している。 $PrevBitCnt$ を前の $RowBitCnt$ と定義する、つまり、 $(d+1)$ の繰り返しの $RowBit$ と定義する。その後、 j は図 29 のように決定される。

【0168】

第3の実施形態を説明するアルゴリズムが図30に表わしてある。このアルゴリズムは第2番目のアルゴリズムと似ており、主要な相違は26～28行目にあるとうり、 j 値前述の記載で計算される。第2の実施形態でのように最初のMBの代わりに j 番目のMBからのデルタにより、MBの $quant$ 値は軽減される。その後、 j 番目のMBは、 $quant$ 値を増加させるために、次のMBに選ばれる。

【0169】

6. 2 再符号化時における不適当なメッセージホールの検出と除去

本発明の他の態様では、再符号化処理の際に、メッセージビットの挿入のために用いるべきでない“不適切な”MHも、決定され、およびまたは、除去される。MHは、以下の4つの理由のために、検出され、およびまたは、除去される。

【0170】

DVDセクタ境界：

もし、ある列のメッセージホールが再符号化後にDVDセクタ境界に存在するとすると、そのMHは検出され、この情報は第2の処理 $GenRmgPck$ に報告される。元のMHは、2つに分かれたMHとして扱われるが、フラグは、その2つのMHが同一のメッセージビットに対応することを示すものとして用いられる。これは、DVDディスクでは、MPEGビデオは、各2KバイトのDVDセクタに分割されるからである。このため、DVDセクタ境界に存在するMHは、2つの異なるDVDセクタに分かれて配置されるので、RM挿入のためのハードウェアのインプリメーションが、さらに複雑になり、コストが高くなる。

【0171】

再符号化エラー：

もし、ある列の再符号化の際に、妥当な回数の試行の後に、MHの付加ビットをその列内の他のMBから補償できないことが分かったら、その列にはMHは挿入されない。さもなければ、再符号化の前に存在した問題、例えばMPEG符号化におけるバッファのアンダーフローやオーバーフローが生じるかもしれない。

【0172】

弱いRMキャリア：

もし、あるMHについての、再符号化後におけるRMキャリアの0と1のビデオデータ間の差が極めて小さいときは、このMHは用いられない。この条件の目的は、RMメッセージビットを回復するための、RMキャリアからの妥当な“信号”強度があることを保証することにある。

【0173】

スキップされたMH：

再符号化の後、もしあるMHがスキップされたMHとして決定されたときは、そのMHは除去される。これは、スキップされたMHからは、RMキャリアに付加されたメッセージ情報は回復できないからである。

【0174】

6. 3 要約：MH候補、RMキャリアおよび再符号化

図31はすでに述べたような、メッセージホール候補とランニングマーキングキャリアを生成するとともに再符号化を行う処理を要約した図である。これらの処理を行う他の方法を選択するために、表8に、4種類の異なる処理モードが定義されている。

【0175】

【表8】

MH処理モード	ファンクション	文字	説明
MHProcMode[0]	MH生成	0	MHをファイル MHin から読み出す。
		1	MHを分類により無作為に生成する。
MHProcMode[1]	再符号化実行	0	再符号化を完全にスキップする。
		1	再符号化をMH付き行用に行用する。
MHProcMode[2]	MHのビットストリーム書き出し	0	MHのビットストリームデータを MHBitOut に書き出さない。
		1	MHの不可ビットストリームを MHBitOut に書き出す。
MHProcMode[3]	MHのビデオデータ書き出し	0	MHのビデオデータを MHVideoOut に書き出さない。
		1	MHのビデオデータを MHVideoOut に書き出す。

様々なMH処理モード

【0176】

上述したように、入力MPEGビットストリームMpegInがMPEG復号

器によって復号され、再符号化に必要な全ての符号化情報が得られる。あるフレームについてMHを生成するために、RMキャリアを生成するために用いられるMBAや他のパラメータが、入力ファイルMHinから読み込まれるか、または上述したようなクラス化によって生成される。このファイルは、MHをランダムに一系列生成した出力ファイル、再符号化後のMHファイル、またはユーザによって編集されたファイルなどである。もし、MHをクラス化によって生成する手法がとられたときは、MHを生成するのに必要なフレーム全体が再構築され、そのフレームについてクラス化が実行される。いくつかのMBに対してRMキャリアを生成するために、DCT係数の代わりにビデオデータを直接変更する手法がとられたときは、これらのMBのビデオデータは再構築される。生成されたMHは、オペレータによって、あるいは再符号化処理を実行することによって、除去される。MHのサイズ、位置および付加ビットストリームは、復号化されるMHのビデオデータとともに、再符号化処理の際に生成され、出力FIFOを用いる再配分プログラムに送られる。MHの再配分処理は、MHの平均個数を、要求されるMHレートに可能な限り近づけるために、行われる。キャリア0とRMGパックを含む最終的なMPEGビットストリームは、第2のプロセスGenRmgPckによって生成される。RMGパックは、MHのサイズ、位置オフセットおよびキャリア1ビットストリームを格納する。

【0177】

図32はMH生成に用いられるGenRmプログラムのアルゴリズムを示している。このプログラムは、まず、ライン1に示すように、入力パラメータファイルからMH再符号化モードを読み出す。このプログラムによって、再符号化モードの値に応じて選択される各MH処理モードが、表8に示されている。それから、ライン4に示すように、画像入力ファイルからの各フレームは復号化され、この復号化情報は記憶される。ライン7に示すように、最初の実行時、MH情報を記憶するためにメモリーが割り付けられる。プログラムは、毎回、シーケンスの最後に達したか否かをチェックする。もしそうであれば、ライン10に示すように、プログラムは再配分処理を呼び出して、残りの処理されたフレームMHを均等に配分する。もし、現フレームがBフレームであり、MHをクラス化けによっ

て生成する手法がとられたときは、ライン16に示すように、プログラムはクラス化プログラムを呼び出して、フレームにMHを生成する。次に、ライン17に示すように、再符号化機能が呼び出されて、MHを含むそのフレーム内の列を再符号化する。再符号化機能は、そのフレームのMHのビデオおよびMPEG情報を出力FIFOに書き込む。セッション内の全てのフレームが処理されたとき、ライン20に示すように、GenRmプログラムは再配分処理を呼び出して、このセッションで処理された全てのフレームにおいてMHを再配分する。

【0178】

再符号化機能は、上述したMPEG再符号化の第3の実施形態を利用する。再符号化処理の反復回数を削減するために、再符号化プログラムは、また、再符号化の際の反復処理で定義された反復状態Inc_delta_Rowを選択することによって、より大きなデルタ（1よりも大きい）の利用を選択することも可能である。再符号化機能は、その列の全てのMBのmquant値を増加させるために、反復状態Inc_delta_RowまたはInc_Mquant_Rowを、選択値デルタか1かによって、それぞれ選択することができる。また、再符号化処理をスピードアップするために、前の列のmquantインクリメントdを、次の列などのmquantインクリメントの初期見積もりとして、用いる。この場合、前の値dが大きすぎるときは、再符号化プログラムは、反復処理で定義された反復状態Dec_delta_Rowを選択することができる。反復状態Dec_delta_RowおよびDec_Mquant_Rowは、その列の全てのMBのmquant値を減らすために、選択値デルタか1かによって、それぞれ選択することができる。

【0179】

再符号化プログラムはまた、再符号化される現フレームについて、その前のイントラまたはノンイントラ量子化マトリックスが変えられるか否かを検出する。この情報は、その後、現フレームの全MHについてMHキャリアを生成する間に、対応するDCT係数に付加されるシグマ値（ノイズ）を、スケールするために用いられる。プログラムはチェックし、MHを含まない列の再符号化をスキップする。再符号化される列の反復状態は、最初は、Inc_delta_Rowに初期化される。その列のMBの元のmquant値は、最初は、前に再符号化された列のm

quant インクリメント d によってインクリメントされる。反復ループの各繰り返しの際に、プログラムは、その列のMBのquant 値を調整することによって、MHの補償を試行する。MBのquant 値の変更に用いられる正確な手法は、反復ループの各繰り返しの際の反復状態の値に依存する。

【0180】

もし、予め設定した最大回数の繰り返し処理の後にも、目標ビットレートが達成できない場合は、その列の全MHは除去され、元の列は置換されない。また、もし、ある列に必要なquant インクリメント d が、予め設定した許容最大歪み値よりも大きくなったときは、その列の全てのMHが除去される。MHに対応するMBに対して、プログラムは、再符号化ループの最初の繰り返しの際に一度だけ、MHブロックをRMとともに、RMなしで生成し記憶する。MHに対応するMBは、それから、特定のメッセージホール置換数（例えば3：オリジナル、キャリア0およびキャリア1）と等しい回数だけ、再符号化される。このそれぞれ再符号化において、対応する符号化ビットストリームは、異なる列バッファに、MH置換子のバイト列とともに記憶される。

【0181】

各MHのキャリア0および1についてのMPEGビットストリームデータは、プログラムにおいて、入力パラメータファイルからの各キャリアのそれぞれの値を読むことによって、選択（制御）される。これらの値は、キャリア0および1についてのMH置換指標値である。このMH置換指標値によって、キャリアのビットストリームデータは、この値に一致する列バッファから選択される。もし、MHのキャリア1と0のビデオデータ間の差が零であるときは、プログラムはこのMHを無効としてマークする。キャリア0として選択された（列バッファの1つから）ビットストリームデータは、常に、各MHに対応するMB内に挿入される。このように、GenRmプログラムによって生成された出力VOBは、各MHについてキャリア0ビットストリームデータを含む。

【0182】

再符号化処理をスピードアップするために、その列のMPEGビットストリームは、最後の繰り返しの際にのみ、ローバッファ（Row Buffer）に書き込まれる

。それ以外は、プログラムは、各繰り返しにおいて、その列を再符号化するために必要なビット数を計数するのみである。また、その列のMBは、最後の繰り返しの際に再び量子化されない、というのは`quant`値は変わらないから。その列のMPEGにおいて、元のサイズ、オフセット、MPEGビットストリームおよびビデオデータは、各MHのキャリア0および1とともに、最後の繰り返しの際にのみ、出力FIFOに書き込まれる。各MHのキャリア1のビットストリームデータは、`MHProcMode[2]`の値が“1”のときにのみ、第2の処理`GenRmgPck`によってファイル`MHBitOut`に書き込まれる。このファイルは、この後に、メッセージビット変調に用いられる。もし、`MHProcMode[3]`の値が“1”のときは、第2の処理`GenRmgPck`がまた、その元の値を、各MHのキャリア0および1の両方のビデオデータとともに`MHVideoOut`に書きこむ。`MHVideoOut`からの情報は、その後に、元のビデオの各MHに挿入された各メッセージビットを回復するために用いられる。プログラムは、その終わりに、ビットスタフピングを実行し、その列の元のサイズと同一サイズを確保する。最後に、プログラムは、再分配を行う現フレームの全ての更新されたMHについて、MH情報（seed、MHAなど）を記憶する。

【0183】

本発明の他の態様によると、再符号化処理をさらに高速化するために、ランニングマークのMPEG2再符号化を、他のCPUおよびまたは他の装置によって、並行して実行してもかまわない。生成された全てのRMを集めて、再生ユニットの動作をシミュレートするための後処理が必要となる。このように、本発明によると、プログラム`GenRmpPck`が、ランニングマークを生成する2段階処理の第2の処理として用いられる。RMが再符号化MPEG2データの形式で生成された後は、プログラム`GenRmpPck`は、そのRMをDVD仕様のVOBに挿入し、ランニングマークVOBを生成する。ランニングマークのキャリア0はビデオに挿入され、キャリア1は`RMG_PCK`セクタに挿入される。この第2の処理は、また、再生ユニットのスループットがリアルタイムでRMを挿入するのに十分であることをチェックする。また、RMを含む新たなVOBが

生成されたとき、プログラムGenRmgPckによって、挿入されたRMの個数を映画の長さに比して評価するチェックがなされる。メッセージのコピーが、プレーヤから送られたメッセージが確実に回復できるように、いくつか補完されるかもしれない。

【0184】

プログラムGenRmgPckは、実行時にアクティブとなる2個の入力モジュールを有する。第1のモジュールは、入力VOBから、セクタのストリームを抽出し特定する。第2のモジュールは、GenRmによって生成されたデータパケットを様々なファイルに読み出して、RMがMPEGでどのように符号化されるかを理解する。メインモジュールは、ランニングマークキャリア0および1を、ビデオパケットとRMパケット(RMG_PCK)に、それぞれ挿入する。その出力は、GenRmから受けたデータから生成された回復ファイルとVOBからなり、RMデータをサポートするために調整された後、一度に1セクタずつ書き込まれる。MPEG2ランニングマークを準備するためにGenRmの複数のコピーを用いるとき、プログラムGenRmgPckは、VOBを完全に処理するために、全てのセグメントを集めてまとめる。もし、ただ1つの処理がRMを生成したとすると、mhdataと名付けられた単一ファイルに全てのデータが格納され、GenRmgPckは必要となる全ての情報をこのファイルから得る。もし、1個よりも多いmhdataファイルが生成されたとすると、GenRmgPckはRMを、ファイルmhdata.000、mhdata.001、mhdata.002など最終ファイル(常にmhdata.999と名付けられる)までからロードする。プログラムはmhdataファイルを読み、最初のRMデータを探す。そして、一致するセクタか空きRMG_PCKを見つけるまで、セクタを出力にコピーする。セクタ一致があると、MPEG2データは置換され、キャリアがRMG_PCKに付加され(RMG_DATAエントリが付加され)、新たなRMがmhdataファイルから読み出される。RMG_PCKがあると、メモリに保持されRMG_DATAフィールドに順次詰められた前のRMG_PCKは、出力ファイルにおいてシーク(Seek())を実行することによって更新され、現RMG_PCKは次のセクションのためにクリアされる。

【0185】

2つの異なるセクタを覆う1個のRMが見つかったとき、GenRmgPckはこのRMを2個のRMに分け、2個のRMG_DATAフィールドを生成する。そのRMの2つの部分は両方とも、RMG_DATAのbcincrモードフィールドを用いることによって同一のメッセージビットを示している。セクタが読まれるたびに、タイミングチェックがなされ、再生ユニットのCPUによるシミュレーションが、データオーバーフローを避けるためにセクタレベルで実行される。各セクタは、そのタイプ(RMG_PCK、ビデオ、その他)やその状態に基づいて、所定の最小処理時間を必要とする。空きRMG_PCKが必要とする時間は、満たされたRMG_PCKよりも少なく、セクタに挿入された各RAMは、実行時間中にデータ置換のためにより多くのCPU時間を必要とする。もし、RMの挿入がプロセッサのオーバーフローを引き起こしたときは、そのRMは除去される。本発明によると、GenRmはまた、元のビデオMBをRMキャリア0および1とともに送る。RMが落ちたとき、元のビデオMBはMPEG2ビデオエレメンタリストリームに挿入され、キャリア0および1は無視される。各RMはメッセージビットと関連しているので、この場合、メッセージビットカウンタは増加しない。

【0186】

さらに、本発明によると、マルチプロセッサコンピュータの処理を高速化するために、MPEG2ビデオストリームを複数の部分に分割するのが好ましい。この場合、特別な配慮が必要になる。というのは、MPEG2符号化は参照フレーム符号化に基づくものだからである。このため、ビデオを適切に符号化するために、少なくとも2つの参照フレームがロードされなければならない。プログラムGenRmおよびGenRmgPckは、DVD VOB内では、ナビゲーションセクタ(NV_PCK)がイントラ符号化フレーム(I)の直前にあるという事実を利用している。図34に示すように、原ストリームは複数のセグメントに分けられる。IはIフレームの前のナビゲーションパックを表し、PおよびBは予測フレーム(第2の種類の参照フレーム)および双方向予測フレームを表している。ハッチの入ったセルは、各セグメントの、GenRmおよびGenRmg

P c kがRMを挿入している部分を表している。

【0187】

図34から分かるように、第1のセグメント以外の全てのセグメントでは、第2のIフレームの前のフレームは、RMを挿入することなく復号化される。なぜなら、これらのフレームはその前のセグメントですでに挿入されているからである。しかしながら、RMが挿入されるとき2つの参照フレームが確実にロードされるように、これらのフレームは復号化されなければならない。

【0188】

6. 4 多重視野角

本発明の他の態様によると、プログラムG e n R mおよびG e n R m g P c kは多重視野角をサポートすることができる。多重角を用いると、再生時に、その映画の部分（セグメント）について、複数の利用可能な角度セグメントから、いずれか1つの視野角（ビデオ）セグメントが再生可能となる。しかしながら、多重視野角をサポートするために、それぞれの角度セグメントのRMの個数は、一定に保たなければならない。これは、各角度セグメントには同一のメッセージビットが挿入されなければならないからである。また、RM（各角度セグメント）は必ずしも同一の位置にある必要はないが、総数は、メッセージ回復が可能なように、同一でなければならない。RMの個数が最小の角度セグメントは、他の角度セグメントのために用いるサイズを与える。RMの個数がより多い角度セグメントについては、メッセージ回復能力を増すために、RMのいずれかを落とすか、重複させる（bcincrモードを用いて）。

【0189】

図33は3種類の可能な角度セグメントを示している。ここでは、元のビデオストリームは、処理のために、セグメントN1、N2およびN3に分割されている。各RMG_PCKが適用される範囲は、RMがセグメント境界と交叉しないように、調整される。このため、様々な角度セグメントに先行する最終バック（RMG_PCK）について、全てのRMは角度セグメントの前にある（セグメントN2の先頭）。同様に、各角度セグメントの最終RMG_PCKは、セグメントN2の最後までにのみMHを含む。このため、図に示すように、N2の各角度

セグメントでは、RMは最初は存在し得ない。というのは、最初の空きRMG_PCKが見つかるまでは、RMを記憶する場所がないからである。同様に、セグメントN3の先頭のビデオには、セグメントN3に最初のRMG_PCKが見つかるまでは、RMは存在し得ない。最後に、プログラムGenRmgPckは全てのビデオセグメントを集めてまとめ、上述したように、VOBを完全に処理する。

【0190】

6. 5 ルックアップテーブルを用いた高速量子化

再符号化の一部として、MBの列の各MBのDCT係数を量子化する。量子化の効果的な処理手法として、MBのDCT係数の全てのあり得る値について量子化を計算し、その結果をルックアップテーブル(LUT)に格納するというものがある。MBの量子化に必要な計算処理は、計算コストがこれよりもはるかに少ないテーブルルックアップ処理に置き換えられる。

【0191】

MBを量子化するために、以下の式が用いられる。

【0192】

$$y = ((32 \cdot x + d / 2) / d) / (2 \cdot mquant)$$

ここで、 x は 8×8 ブロックの各画素のDCT係数値、 d は画素の量子化行列係数、そして $mquant$ は、 8×8 ブロックをどの程度粗くまたは精密に量子化するかを定める量子化スケール係数である。この式は、MBの各 8×8 ブロックについて、輝度成分および色差成分の両方について計算される。ただし、量子化行列は、輝度値と色差値とで異なる。 8×8 ブロックの64個の x 値のそれぞれについて、 y 値が計算される。再符号化でこの式が演算される回数は多いので、この式の演算を減らすことによって、実質的な計算量を削減することができる。

【0193】

上式の y 値は、幅広い値をとり得る3個の整数値変数(x 、 d および $mquant$)から求められるため、この式を3次元(3インデックス)LUTにそのままインプリすると、非現実的なほど大容量の記憶媒体が必要になる。このような記憶容量の要求を軽減するために、上式を2つの部分に分けて、2個のLUTに

インプリすることができる。その2つの式は、次のようになる。

【0194】

$$y1 = (32 \cdot x + d / 2) / d$$

$$y2 = y1 / (2 \cdot mquant)$$

ここで、 $y2$ は上述の y と同一である。2次元(2インデックス) LUTが、この2つの式をそれぞれインプリメントするために用いられる。最初の式に係るLUTすなわちLUT1は、 x および d をインデックスとする2次元(行列)テーブルである。第2の式に係るLUTすなわちLUT2は、LUT1から検索した値と、 $mquant$ をインデックスとして用いる。LUT2は $y2$ (そして y)のとり得る値の全てを格納する。

【0195】

LUT1について考える。 x の値は-2048から2047までである。もし、例えば d が1のときは、 $y1$ の値は $32 \times (-2048)$ から 32×2047 まで(-65536から65504まで)の範囲となる。 $y1$ の範囲が大きく、-65536と65504の間の中間値の多くは生じ得ないので、 $y1$ の値をLUT2のインデックスとして直接利用することは(記憶容量の点で)効率的でない。この理由から、 $y1$ の値をLUT1に格納する代わりに、他の種類の整数値すなわち $yindex1$ が格納される。この $yindex1$ は、0からある最大値までの連続した整数値となる。 $y1$ の各値について、唯一の $yindex1$ が存在する。言い換えると、 $yindex1$ と $y1$ とは、1対1の対応関係がある。 $yindex1$ 値は、LUT2に適切な(効率的な)インデックスを与える。

【0196】

要約すると、量子化式を1組の2次元LUTにインプリメンテーションすることは、次のように表される。

【0197】

$$yindex1 = LUT1[x, d]$$

$$y2 = LUT2[yindex1, mquant]$$

ここで、 $LUT1[x, d]$ は、 x および d をインデックスとして用いてLUT1から検索した値を表す。

【0198】

結論

述べられたのは、圧縮されたデジタル信号ストリーム、例えばMPEG標準でエンコードされたビデオストリームの、好ましくは目立たないようにするために画像の加工された部分に配置され、また、フレームごとに位置を変えるピクセルブロックまたはマクロブロックの形式の中にマークを含むための変換のための方法論である。マークのロケーションは、オリジナルのビデオ（または他の情報）とともに参照媒体に保存される“メッセージホール”によって定義される。マークは、DVDやテスト媒体などのテスト下の媒体中のメッセージホールに対応する内容を比較することによりデコードされる。マークは、テスト下の媒体が複製されたソースを示す情報、オリジナル媒体や、複製が行われたプレイバックユニットのシリアルナンバー、および複製の時刻などを表す。この情報は、複製者による検出や改変を防ぐために暗号化され、スクランブルされ、好ましくは、多数のソースの識別能を高め、ノイズがあっても復調を可能にするために展開される。

【0199】

ここでは本発明のベストモードが述べられたが、本発明は異なるようにも実践され得る。例えば、本発明は事前に記録された媒体の保護の環境について説明されたが、本発明は同様にインターネットや他のコミュニケーション媒体から得られるデータに対しても適用できる。他の変形としては、本発明は、述べられたようなムービーの複製のソースを特定するだけでなく、ディスクや他のタイプの媒体、またはネットワークから複製されたソフトウェアにも適用できる。さらに、開示は保護されたデータの複製者の追跡の例を用いてなされたが、本発明は複製を防ぐためのメッセージデータを用いるためにも適用できる。例えば、本発明は、保護された素材の判読可能な複製がなされ、または有用なプレイバックが行われるのを防ぐためにランニングマークメッセージが適用されるウォーターマークコピー防止のコンテキストに適用され得る。さらに、ランニングマークメッセージは、望まれるなら、機関によって発行されて、顧客にディスクまたは他の形式のメディアまたはネットワーク空のデータの正当な複製を認可する認可コードと

比較され得る。この認可は、例えばDVDセクタをプログラムしたりまたは、顧客がDVDを1回限りまたは複数回再生し、または所定の数の複製を作製するのを可能にするための他の保護の実現をなすために適用され得る。セクタは、別の方法として、顧客がDVDを種々の目的に用いたり、無制限の使用を可能にしたりするのを防ぐためにプログラムされ得る。

【図面の簡単な説明】

【図1】

本発明によるMPEGランニングマーク位置の作成を示す高レベル機能ブロック図である。

【図2】

コピーのソースメッセージを行う3つのメッセージホールのある画像を示す。

【図3】

ビデオストリームで実施したメッセージホールを示す線図である。

【図4】

(a) および (b) は、本発明によるランニングマークシステム符号器及び復号器を示すブロック線図である。

【図5】

データ記号基準を使用したメッセージビットの記号化を示す線図である。

【図6】

(a) および (b) は、鎖状誤差訂正符号化及び複合化の線図である。

【図7】

本発明の実施例で実行する包旋状誤差訂正コードを示す。

【図8】

本発明の実施例で実行するCDMA符号化処理を示す線図である。

【図9】

(a) - (e) は、ビデオストリームで実施したコピーのソースメッセージのセキュリティを向上させるためのマルチプル可変長配行を示す。

【図10】

(a) - (d) は、本発明によるビット符号化の変形例を示す

【図11】

本発明に一形態でのDCTキャリア係数へのノイズ組成を加えることによる元の信号波形の広がりを示す。

【図12】

本発明におけるRMカメラ生成のためのDCTの試みを示す擬似コードである。

【図13】

DCTの試みを使用した輝度ブロックのためのピクセルドメインにおけるノイズパターンを示す。

【図14】

(a) - (d) は、ランニングマークデータストリームのレイアウトと構成の線図である。

【図15】

本発明によるDVD信号ストリームで行われるビット抽出を示す線図である。

【図16A】

本発明によるランニングマーク生成の動作を示すアルゴリズムである。

【図16B】

本発明によるランニングマーク生成の動作を示すアルゴリズムである。

【図17】

本発明の一形態によるDivxマーク挿入前、挿入後のビデオグループのレイアウトを示す。

【図18】

本発明の一形態によるDivxマーク挿入前、挿入後のビデオグループのレイアウトを示す。

【図19】

gMH行を定義するアルゴリズムである。

【図20】

本発明に一形態によるマクロブロックの分類のための擬似コードである。

【図21】

分類機能によって要求される初期化 () のための擬似コードである。

【図 2 2】

マクロブロック分類の他の実施例のための擬似コードである。

【図 2 3】

処理する典型的なフレーム区分の線図である。

【図 2 4】

再分配手順の擬似コードである。

【図 2 5】

再分配機能によって要求される初期化 () のための擬似コードである。

【図 2 6】

メッセージホールの予備ビットの補間するための流れ行におけるマクロブロックのための M_{quant} の増加数値を示す。

【図 2 7】

本発明による MPEG 再符号化の第一形態を示すアルゴリズムである。

【図 2 8】

第 2 形態を示すアルゴリズムである。

【図 2 9】

k 回 ($k = (d + 2)$) 繰り返すための j の位置決めアルゴリズムである。

【図 3 0】

MPEG 再符号化の第 3 形態を示すアルゴリズムである。

【図 3 1】

本発明によって行われる機能をまとめた高レベル線図である。

【図 3 2】

GenRM プログラムを示すアルゴリズムである。

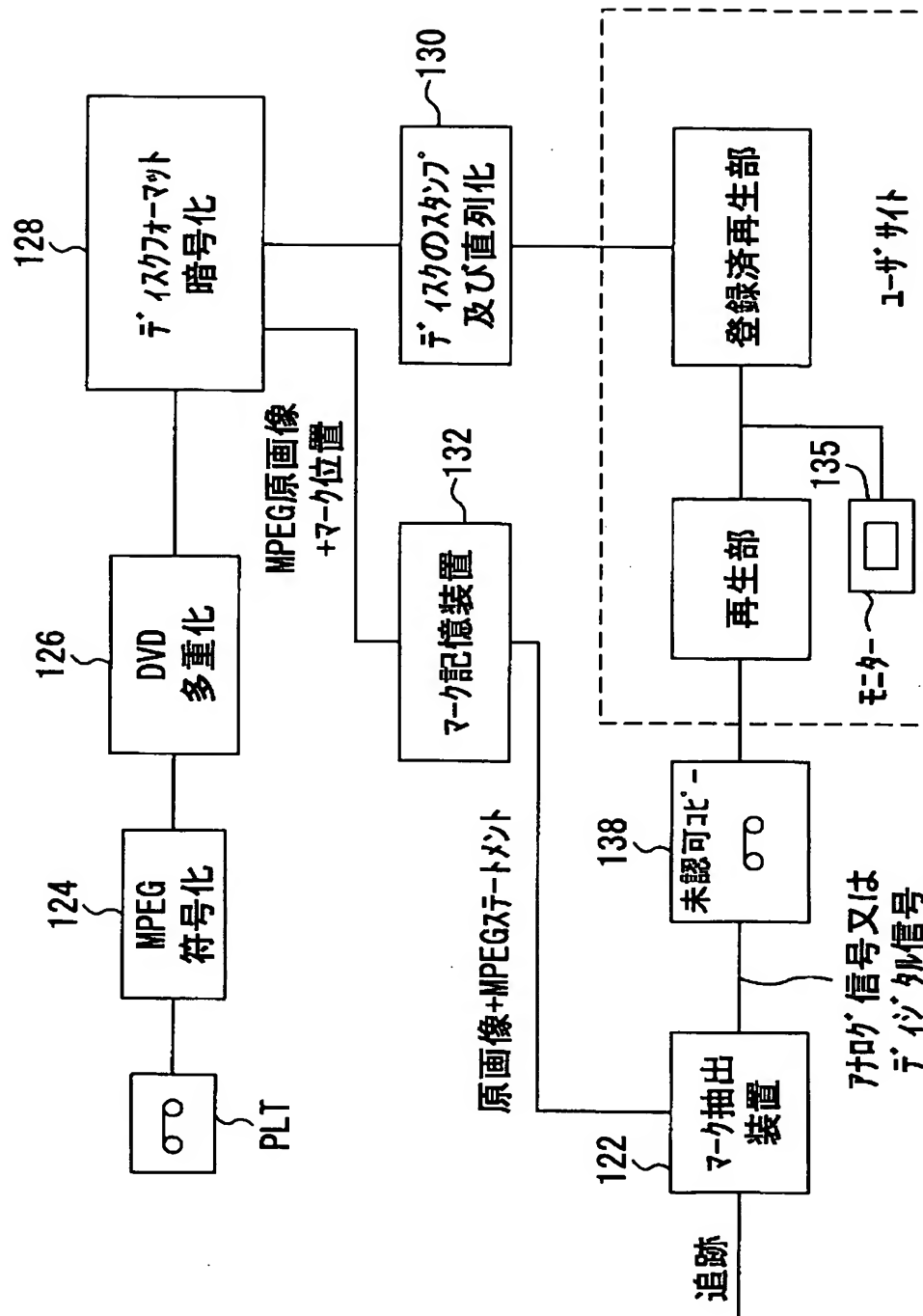
【図 3 3】

見る角度を変えたビデオ表示を示す。

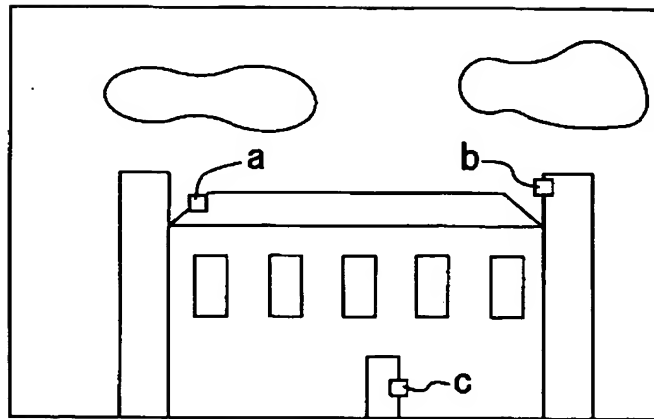
【図 3 4】

複数のセグメントを示す。

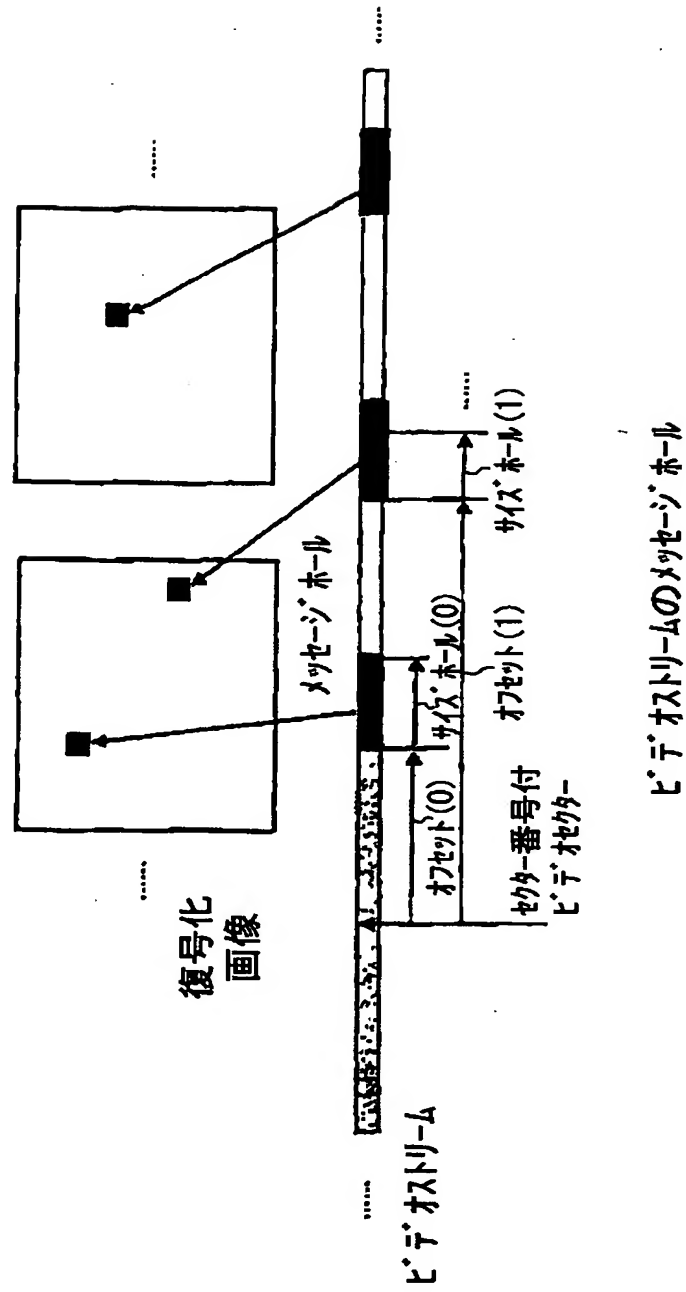
【図1】



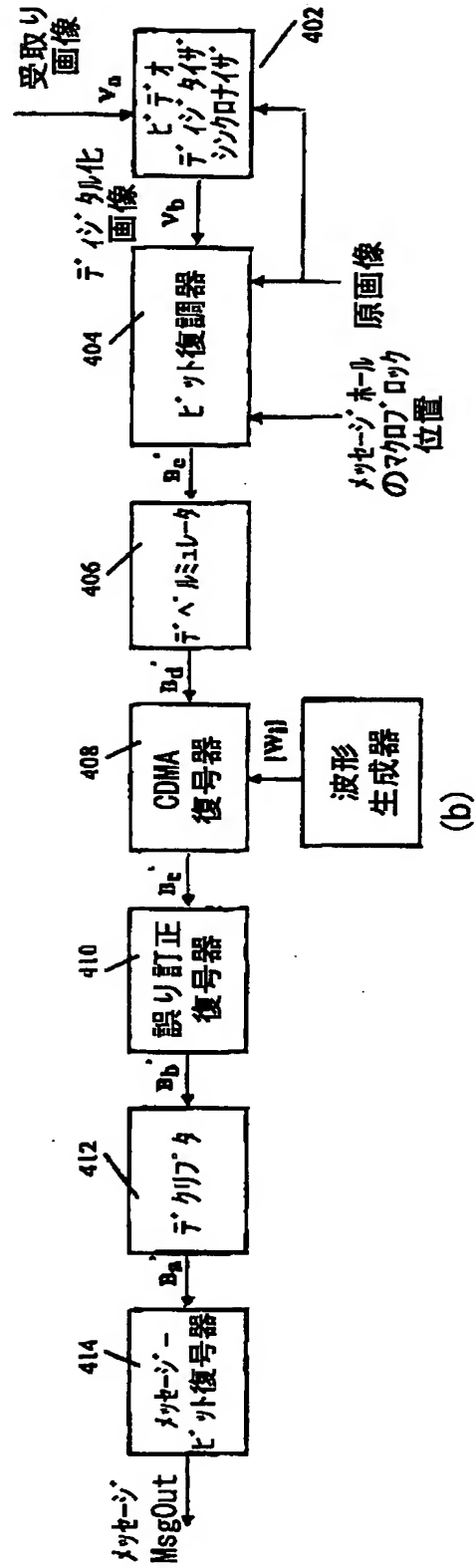
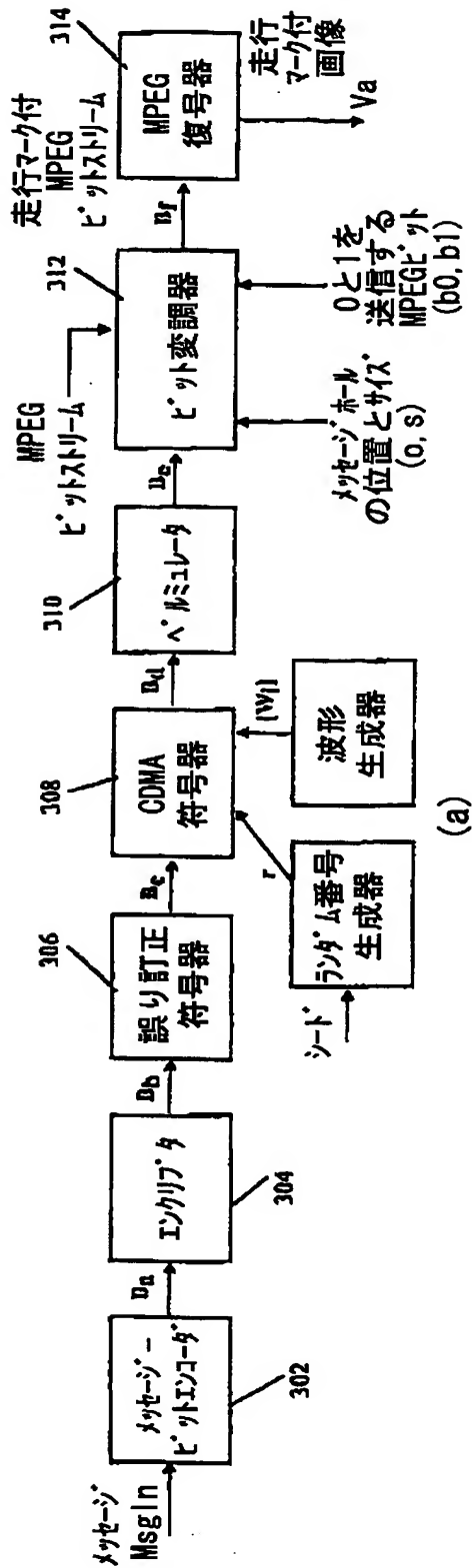
【図2】



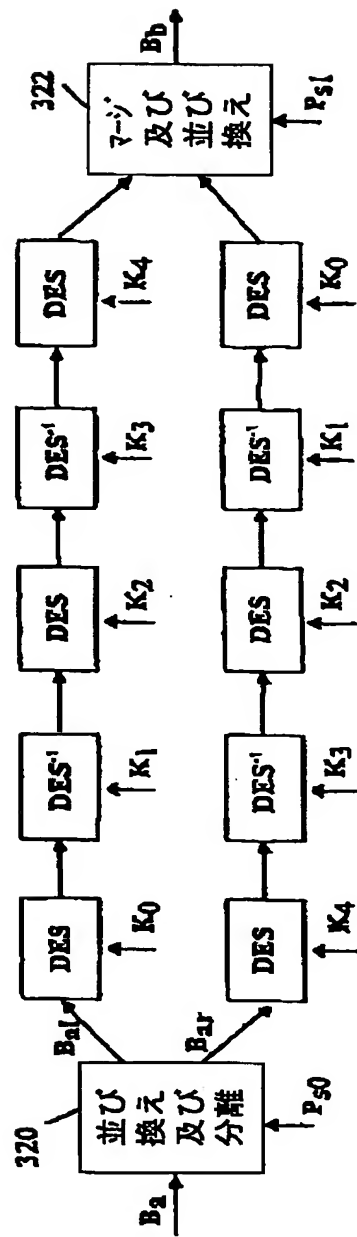
【図3】



【図4】

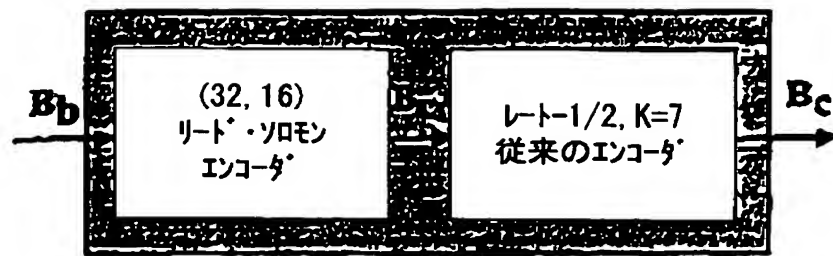


【図5】

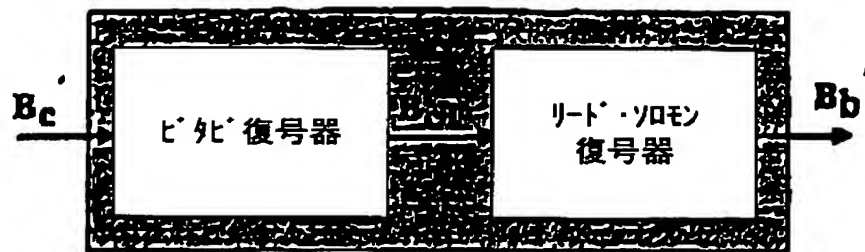


128ビット前並び換え及び後並び換えでデータ暗号化標準
(DES)を用いたメッセージビットの5重暗号化

【図6】

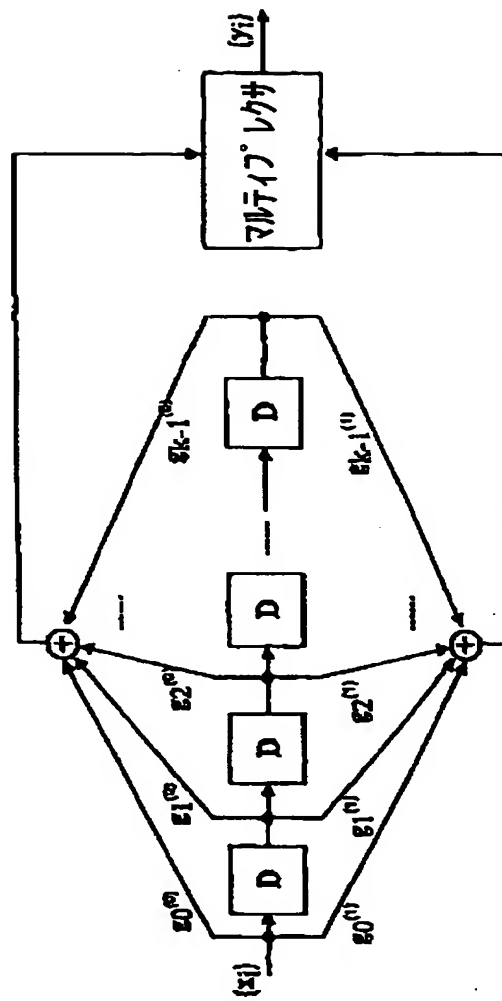


(a)



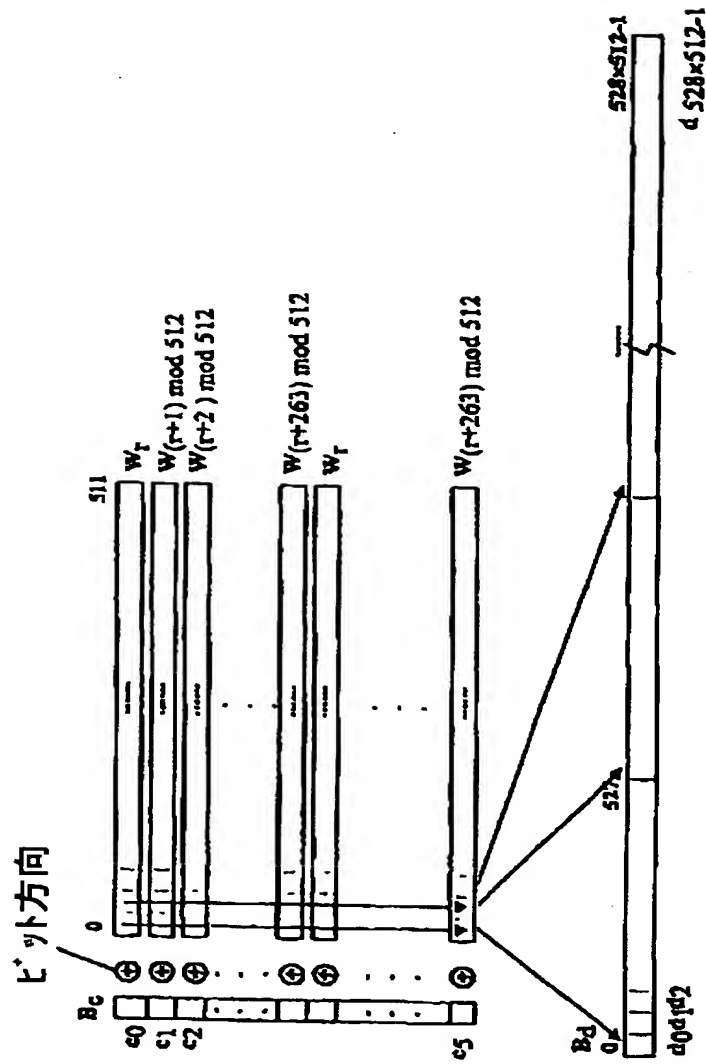
(b)

【図7】



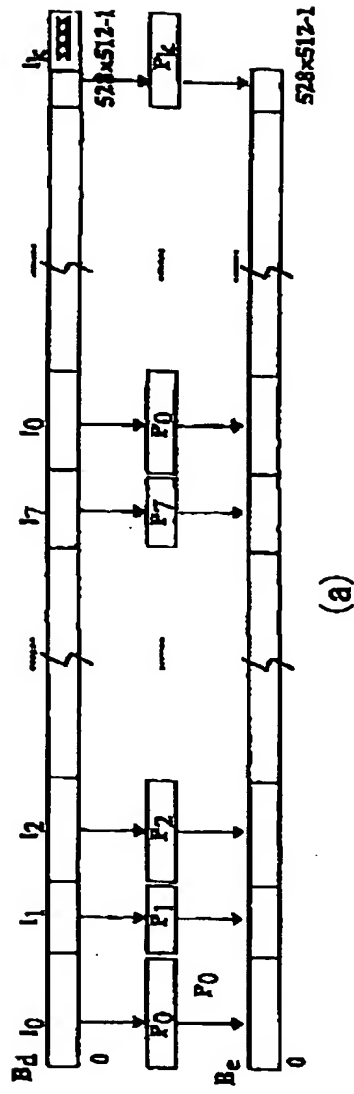
拘束長Kの従来の1/2レート誤り訂正符号器

【図8】

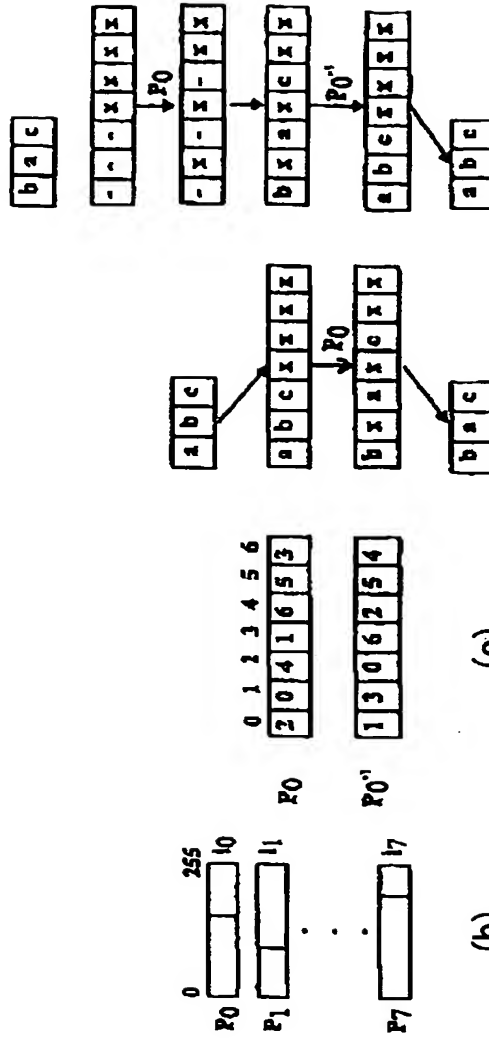


CDMA符号化処理: W_r は無作為に抽出された波形

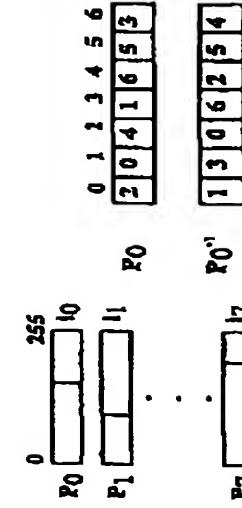
【図9】



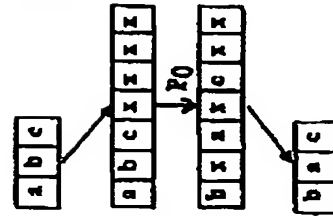
(a)



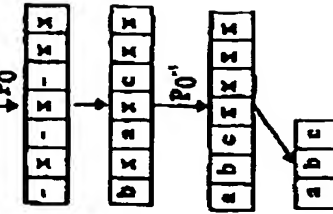
(b)



(c)



(d)

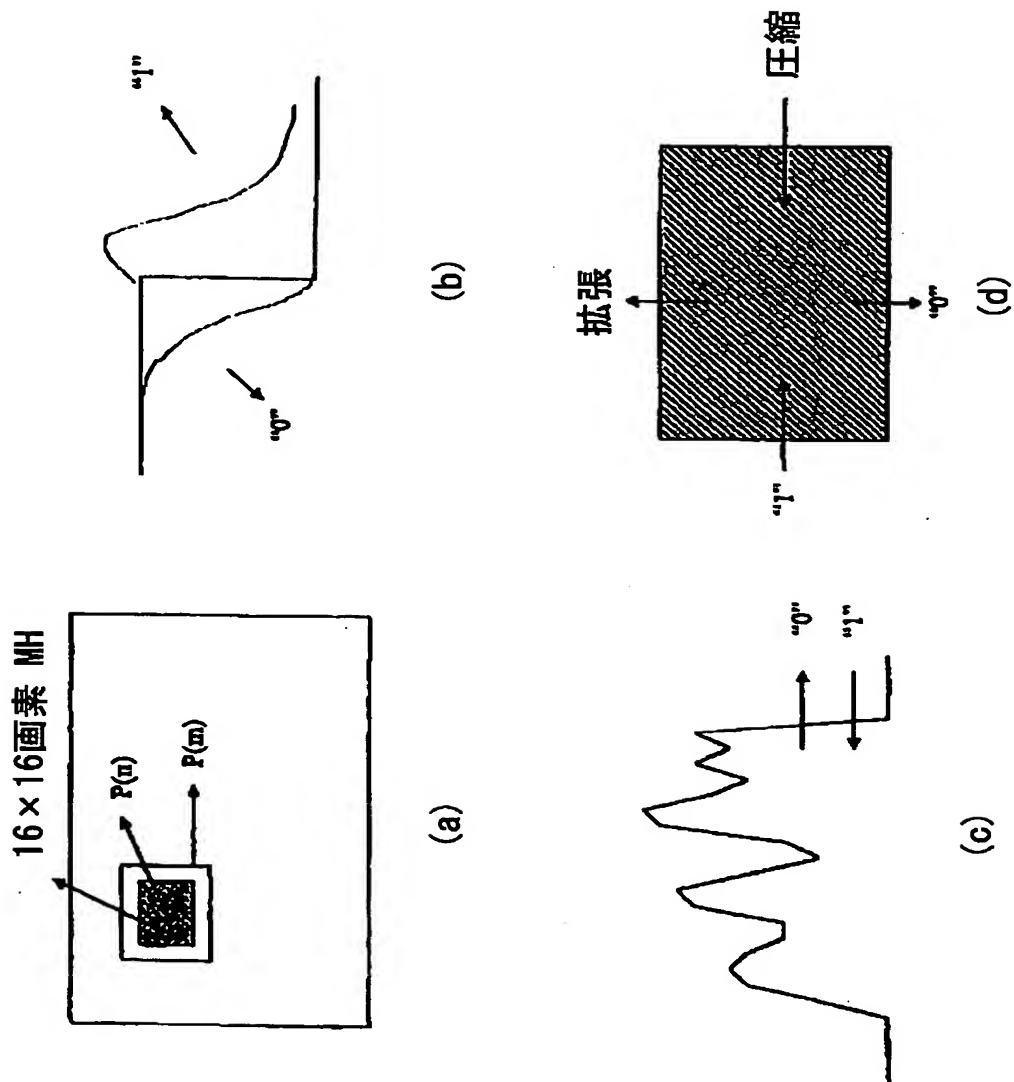


(e)

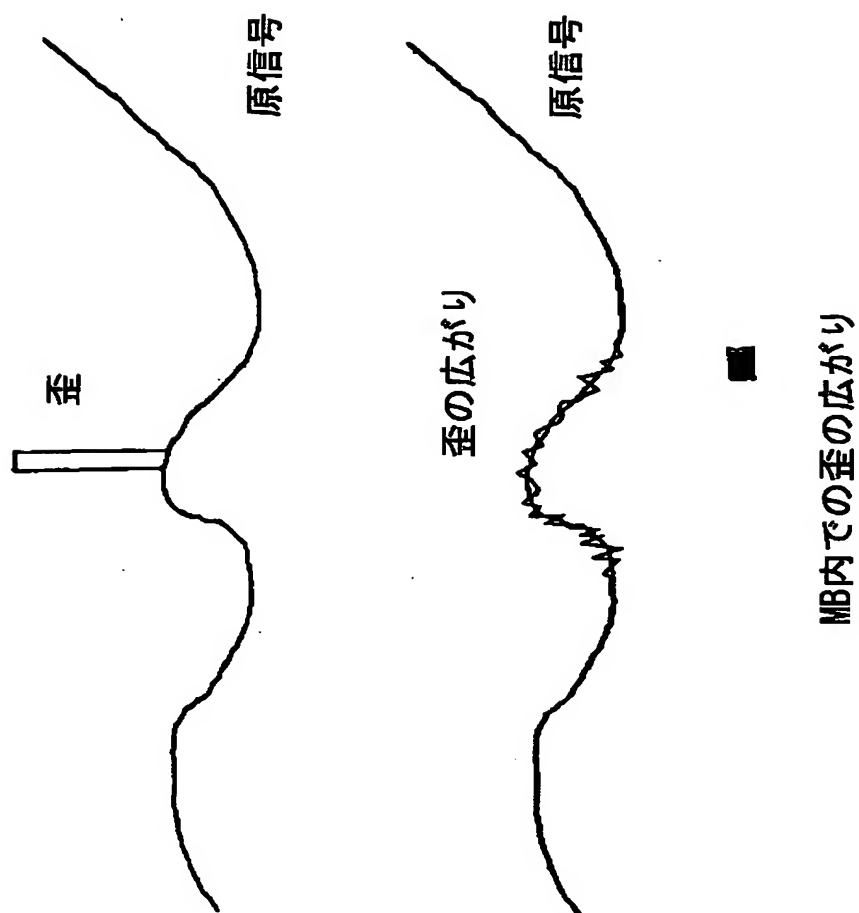
(置換子)

多重可変長並び換え: (a) 入力ビット処理、(b) 8種類の長さパルミュータ
 (c) P_0 とその逆数の例、(d) P_0 を用いたビット最終ブロックの処理、(e) ビット最終ブロック復元

【図10】



【図11】



【図12】

```

1  for all the blocks of a MH
2      Assign all the DCT coefficients to the Mhblock[0][block]:
3
4  Initialize the random number generator with an input seed number:
5
6  for all the blocks of a MH {
7      Select nDCT and sigma for luminance and chrominance blocks:
8      Generate Arr[ ] of nDCT elements randomly with values of 0, 1 and 2
9          according to the input ratio:
10
11     for(i = 0; i < 64; i++){
12         if( 1 <= i <= nDCT){
13             if( Arr[i] == 1 ) {
14                 Mhblock[1][block][i] = Mhblock[0][block][i] + sigma;
15                 Mhblock[2][block][i] = Mhblock[0][block][i] - sigma;
16             }
17             else if ( Arr[i] == 2 ) {
18                 Mhblock[1][block][i] = Mhblock[0][block][i] - sigma;
19
20                 Mhblock[2][block][i] = Mhblock[0][block][i] + sigma;
21             }
22             else {
23                 Mhblock[1][block][i] = Mhblock[0][block][i];
24                 Mhblock[2][block][i] = Mhblock[0][block][i];
25             }
26         }
27         else {
28             Mhblock[1][block][i] = Mhblock[0][block][i];
29             Mhblock[2][block][i] = Mhblock[0][block][i];
30         }
31     }
32     Bound Mhblock[1][block] and Mhblock[2][block] into legal DCT range:
33 }

```

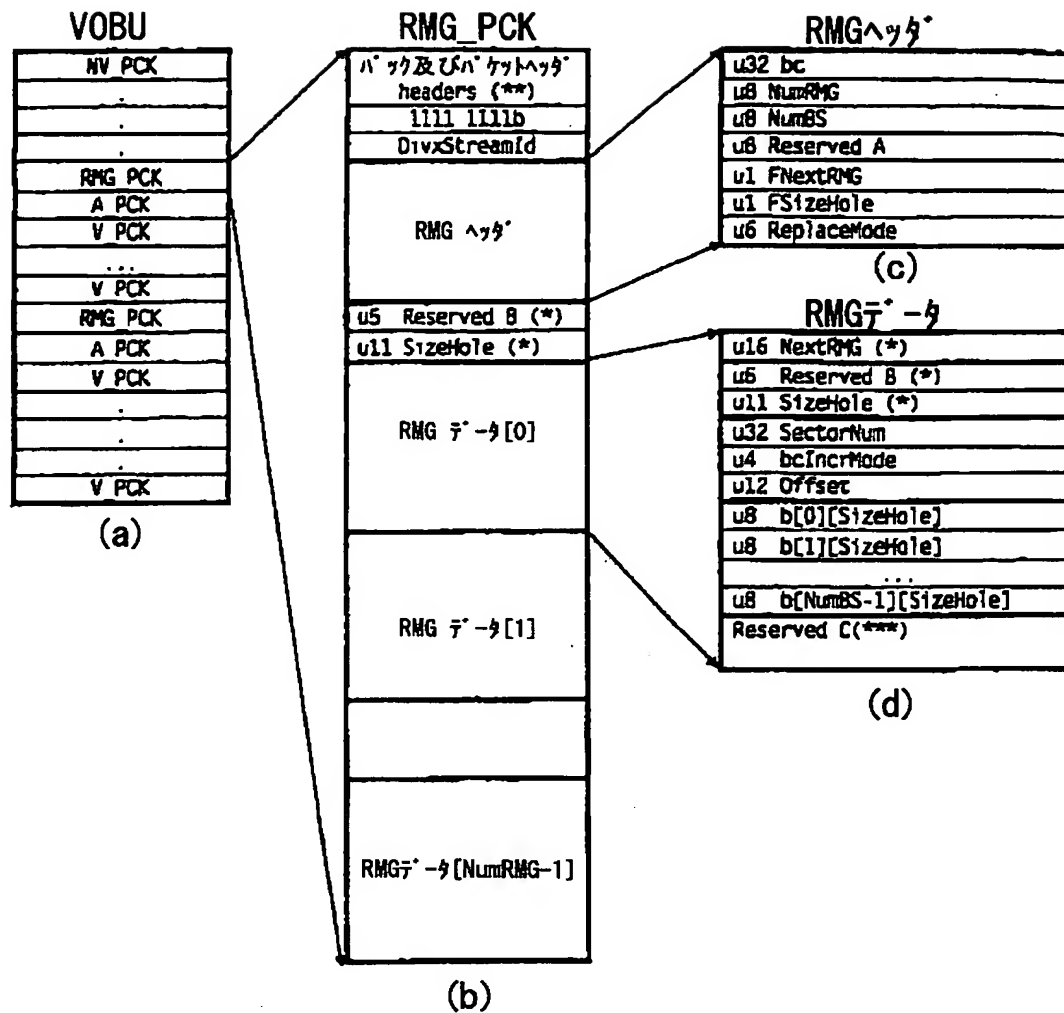
RMキャリアを生成するDCT法

【図13】

0	4	-1	-1	-7	-6	2	-1	9	0	-1	5	7	-4	3	0
0	-3	-2	5	0	0	4	-1	9	0	-1	-2	0	-1	2	1
1	-2	-1	1	3	0	3	2	1	-5	1	-2	-3	-3	-3	0
2	3	-1	-2	-1	-3	2	5	1	-5	1	1	1	-3	-4	3
1	1	7	1	-2	-4	2	2	9	1	-4	-1	3	5	2	4
1	1	5	-3	1	0	-1	-2	2	1	-4	-4	-1	5	0	-1
4	1	0	-9	5	2	-2	3	-6	-4	-2	-5	-4	3	-5	-1
4	-6	-3	-8	3	1	-1	0	1	-4	-2	-4	-4	6	-5	6
-2	3	6	-5	-3	0	3	-2	2	-2	-5	3	2	2	-3	-4
6	-1	0	-7	1	-7	0	-2	2	0	-3	4	8	3	-2	0
1	-4	-4	0	8	-5	1	0	-3	3	4	2	1	-2	5	4
-6	0	0	7	9	-1	5	2	-7	0	4	0	-5	-8	4	0
-2	5	1	6	3	0	4	0	-4	-2	-3	-1	3	-7	-5	-3
-2	-2	-1	1	5	-2	2	-4	2	0	0	-1	-2	-4	-3	1
-5	-4	-2	1	4	0	-1	-5	1	2	7	-1	-9	0	3	5
0	3	2	1	-4	0	-3	-3	-3	1	6	1	-3	6	-2	4

nDCT=40、sigma=5及び0, 1, 2の割合と2:1:1とするパラメータにより
DCT法を用いた輝度ブロック用画素領域の代表的ノイズパターン

【図14】



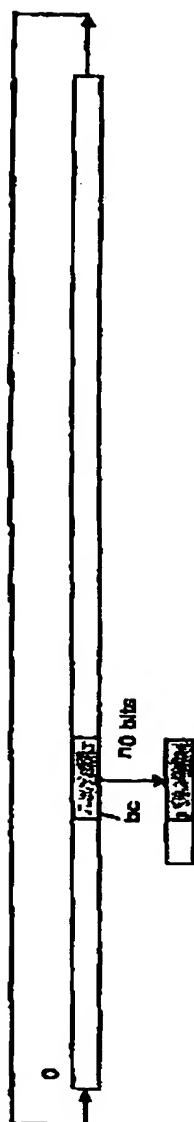
走行マーカー・ストリームのデータ配置及びデータ構造: (a) VOB内RMG_PCKのデータ配置、(b) RMG_PCKのデータ配置、(c) RMGヘッダ、(d) RMGデータ[0]

★印の変数は存在しない場合有り

★★: パケット及びパケットヘッダはV15-9のセクション5, 2でDVD仕様に従う

★★★: この予約済領域はFNextRMG=1でない限り存在しない。FNextRMG=1の場合、NextRMGは次のRMGデータの配置用に使用される。

【図15】



Bits[]から送信用ビットの抽出

【図16A】

```

0  /* For: An algorithm of run-time module for a running mark system. */
1  /* By:  Siu-Leong Iu & Guillaume Mercier of Circuit City Stores Inc.. 4/28/98 */
2
3  int RunTimeModule(){
4      int i;
5      int errorNum = NoError;
6      u8 StreamId, SubStreamId, DivxStreamId;
7
8      while(readSector() != EndOfDVD){
9          errorNum = RunningMarking();
10         ErrorAction(errorNum);
11         readFromSector StreamId;
12         switch(StreamId){
13             case VideoStream:
14                 Decode MPEG video bitstream and send result to video port;
15                 break;
16             case AudioStream:
17                 Decode MPEG audio bitstream and send result to audio port;
18                 break;
19             case PrivateStream1:
20                 readFromSector SubStreamId;
21                 switch(SubStreamId){
22                     case ProviderStream:
23                         readFromSector DivxStreamId;
24                         switch(DivxStreamId){
25                             case RMGStream:
26                                 readFromSector bc, NumRMG, NumBS, FVextRMG, FSizeHole, ReplaceMode;
27                                 if(!FSizeHole)
28                                     readFromSector Reserved B. SizeHole;
29                                 for(i=0; i<NumRMG; i++)
30                                     ReadFromSector RMG_Data[i];
31
32                                 break;
33                             case OtherDivxStreams:
34                                 break;
35                             }
36                             break;
37                             case OtherSubStreams:
38                                 break;
39                             }
40                             break;
41                             case PrivateStream2:
42                                 Process private stream 2;
43                                 break;
44                             }
45                             return(errorNum);
46                     }
47

```

走行マークシステム用RMGランタイムモジュールのアルゴリズム

【図16B】

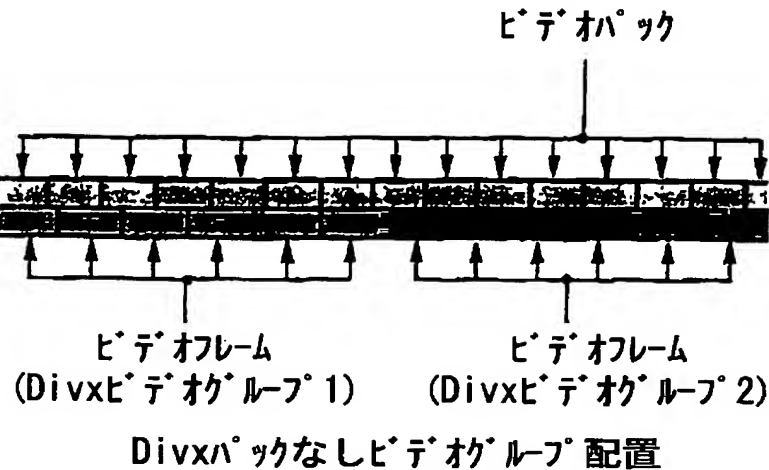
```

48  int RunningMarking(){
49      int i,j;
50      int errorNum = NoError;
51
52      for(i=0; i<NumRMG; i++){
53          if(FNextRMG) NextRMG = RMG_Data[i].NextRMG;
54          else calculate NextRMG;
55          if(FSizeHole)
56              SizeHole = RMG_Data[i].SizeHole;
57          SectorNum = RMG_Data[i].SectorNum;
58          if(SectorNum > CurrentSectorNum)
59              break;
60          bcIncrMode = RMG_Data[i].bcIncrMode;
61
62          if(SectorNum == CurrentSectorNum){
63              Offset = RMG_Data[i].Offset;
64              for(j=0; j<NumBS; j++){
65                  b[j] = RMG_Data[i].b[j];
66                  ReplaceMessageHoleData(ReplaceMode, bc);
67              }
68              update bc by bcIncrValue using bcIncrMode;
69          }
70
71      return(errorNum);
72  }
73
74  void ReplaceMessageHoleData(u6 Mode, u32 bc){
75      u16 sendbit =0;
76      locate MH bits of Message Hole using SectorNum and Offset;
77      switch(Mode){
78          case 0: /* Replace MH bits with b[0] */
79              replace MH bits with b[0];
80              break;
81          case 1: /* Replace MH bits with b[sendbit-1] */
82              extract n0 bits from Bits[ ] pointed by bc and store them to sendbit as in
83              figure 4;
84              if(0< sendbit && sendbit <= NumBS)
85                  replace MH bits with b[sendbit-1];
86              break;
87          case 2: /* Replace MH bits with b[reverse(sendbit)-1] */
88              extract n0 bits from Bits[ ] pointed by bc and store them to sendbit as in
89              figure 4;
90              reverse each one of n0 least significant bits of sendbit;
91              if(0< sendbit && sendbit <= NumBS)
92                  replace MH bits with b[sendbit-1];
93              break;
94          case OtherVersions:
95              break;
96      }
97  }

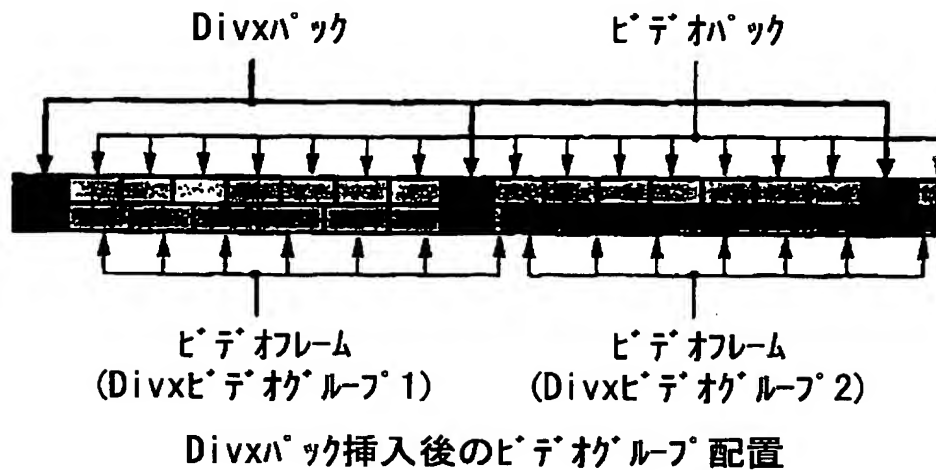
```

走行マークシステム用RMGランタイムモジュールのアルゴリズム

【図17】



【図18】



【図19】

```

1  Obtain values for the Width, Height, nFrames, numMRow, and seed parameters;
2
3  MB_Width = Width / 16;
4  Make sure the numMRow parameter is in a range of 0 to MB_Width;
5  Initialize the random number generator with the value of the seed parameter;
6
7  for (each frame in a total of nFrames frames) {
8      Clear (empty) the list of candidate MB addresses, MBAList;
9      for (each MB row of the current frame) {
10         lowMBA = the MBA of the first MB in that row;
11         highMBA = the MBA of the last MB in that row;
12         while (less than numMRow MBs have been selected on this MB row) {
13             do {
14                 randMBA = a pseudo-random integer between lowMBA
15                             and highMBA;
16             } while (randMBA is found in MBAList);
17             Append randMBA to MBAList;
18         } /* next MB row */
19         Sort the MBA in MBAList into ascending order;
20         Output the entire MBAList;
21     } /* next frame */

```

gMRowのアルゴリズム

【図20】

但し

MHavRow[i]: 行iに使用可能なMH数
 MHallocRow[i]: 行iに割り合てるMH数
 MBtried: 検索されたMB数
 numMHreq: フレーム当たりの要求されたMH数
 maxMHallow: 1行内の最大可能MH数

```

1  initialize();
2
3  while (      the no. of MHs requested have not been obtained and
4              the number of MBs examined is less than the maximum limit)
5  {
6      thisMBA = a randomly selected MBA of a MB that is available;
7      thisMB = the MB located at thisMBA;
8      MBrow = the MB row in which thisMB lies;
9      MBcol = the MB column in which thisMB lies;
10
11     if (most pixel values of thisMB are not within an acceptable range) {
12         Mark thisMB as unavailable;
13     } else {
14         class_type = Classify_MacroBlock(thisMB);
15         if (class_type is TEXTURED) {
16             if (MBrow doesn't already have the maximum no. of MHs per row on it)
17             {
18                 Mark thisMB as picked;
19                 Mark the MHspace MBs on both sides of thisMB as
20                 unavailable;
21             }
22         } else if (class_type is SMOOTH) {
23             Mark thisMB as unavailable;
24         }
25     }
26 }

```

MB分類用擬似コード

【図21】

initialize()

```

1   for (all MBs in the image) {
2       Mark them as available;
3   }
4
5   for (all MBs selected in the previous frame) {
6       Mark them as unavailable;
7   }
8
9   for (j = all the rows in the image) {
10      if (this row is the first or last row) {
11          for (all the MBs in the row) {
12              Mark them as unavailable;
13          }
14      } else {
15          for (all MBs within BorderLimit distance from the border) {
16              Mark them as unavailable;
17          }
18      }
19  }

```

分類ルーティーンにより呼び出されるinitialize()用擬似コード

【図22】

Classify MacroBlock():

Notations used:

Ev - Vertical Energy
 Eh - horizontal energy
 Ed - diagonal energy
 Ea - Average energy

```

1   for (each of the 4 luminance 8x8 blocks in the MB) {
2       dct = the matrix of DCT coefficients of the block;
3
4       Eh      = (dct[0][2]2 + dct[0][3]2 + dct[0][4]2
5               + dct[1][2]2 + dct[1][3]2 + dct[1][4]2)/6;
6       Ev      = (dct [2][0]2 + dct[3][0]2 + dct[4][0]2
7               + dct[2][1]2 + dct[3][1]2 + dct[4][1]2)/6;
8       Ed      = (dct[0][1]2 + dct[1][0]2 + dct[1][1]2
9               + dct[2][2]2 + dct[2][3]2 + dct[3][2]2 + dct[3][3]2 )/7;
10      Ea = (Eh + Ev + Ed)/3;
11      if (Ea > T1)
12          Classify the 8x8 block as TEXTURED;
13  } /* next block */
14
15  if (3 or 4 blocks in the MB are classified to be TEXTURED)
16      The MB type is TEXTURED;
17  else
18      The MB type is SMOOTH;

```

MB分類用擬似コード

【図23】



代表的セクションの図

【図24】

rebMHfn():

Notations used:

TotalMHsize - sum of the size of the MHs that have been selected;

MaxMHsize - the maximum total size of all MHs allowed in a section;

MHsize - the size of the selected MH;

```

1  initialize();
2
3  while (      there are MHs in the section that have not been selected
4              and TotalMHsize < MaxMHsize)
5  {
6      for (thisFrame = all B frames in the section) {
7          Process the first frame every 1/startIncr times through this loop;
8          Process the last frame every 1/endIncr times through this loop;
9          if (there are MHs in thisFrame that have not been selected) {
10             selectedMH = the smallest MH of the unselected ones in thisFrame;
11             MHsize = the size of selectedMH;
12
13             if (TotalMHsize + MHsize <= MaxMHsize) {
14                 Add MHsize to TotalMHsize;
15                 Select selectedMH as a final MH;
16             } else {
17                 Break out of (terminate) the while loop;
18             }
19         }
20     }
21
22     for (all MHs selected during the last iteration of the preceding while loop) {
23         deselect the MH;
24     }
25
26     step = round(no. of B frames per section / no. of MHs deselected);
27
28     for (thisFrame = every 'step' B frames in the section) {
29         if (there are MHs in thisFrame that have not been selected) {
30             selectedMH = the smallest MH of the unselected ones in thisFrame;
31             MHsize = the size of selectedMH;
32             if (TotalMHsize + MHsize <= MaxMHsize) {
33                 Add MHsize to TotalMHsize;
34                 Select selectedMH as a final MH;
35             }
36         } else {
37             while (all B frames in the section have not been searched) {
38                 Search on both sides of thisFrame;
39                 if (a frame is found with an unselected MH) {
40                     selectedMH = the smallest MH of the unselected ones in
41                     thisFrame;
42                     MHsize = the size of selectedMH;
43                     if (TotalMHsize + MHsize <= MaxMHsize) {
44                         Add MHsize to TotalMHsize;
45                         Select selectedMH as a final MH;
46                     } else {
47                         Break out of (terminate) the while loop;
48                     }
49                 }
50             }
51         }
52     }
53     Output all the MHs picked by this algorithm.

```

再配分用擬似コード

【図25】

Initialize()

Notations used:

```

1  if ( the      last      B      frame      of      the      previous      section      is
2    {          the first B frame of the current section)

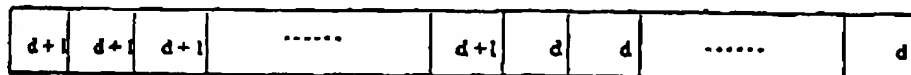
3      Add 1 to the no. of B frames to be processed;
4  }
5
6  for (all B frames in the section) {
7      for (all MHs in the B frame) {
8          if (the MH is on a DVD boundary) {
9              Mark this MH as unavailable;
10         }
11     }
12 }

```

再配分関数により呼び出されるinitialize()用擬似コード

【図26】

現在行のMB: 0 1 2 j+1 j+2



MH符号の余剰ビットを補償する現在行MB用Mquant値の増加

【図27】

```

1  for ( each row in the current frame ){
2    if ( this row is a MH row){
3      iterStatus = Inc_Mquant_Row;
4      for (iterCnt = 0; iterCnt <= IterMax; iterCnt++){
5        if ( ( iterCnt != 0 ) and (iterStatus != Final_Iteration) ){
6          Increase the mquant value of selected MB by 1;
7        }
8        Re-encode the whole row using the current mquant values for each MB;
9        if ( iterStatus == Final_Iteration )
10         Store the encoded bit stream for the whole row into the Row Buffer;
11       else
12         Only count the number of bits needed to re-encode row;
13       Byte Align the number of bits needed to re-encode row;
14       if (iterStatus == Final_Iteration){
15         Write video and mpeg information for MHs in the row to output Fifo;
16         Exit the iteration loop;
17       }
18       else if ( BitsDiff <= 0 )
19         iterStatus = Final_Iteration;
20       else
21         Select the next MB;
22     }
23     Perform bit stuffing to make OrigNumBits = RowBitCnt;
24     Replace original row with the encoded bits from Row Buffer;
25   }
26 }

```

第1のMPEG再符号化方法のアルゴリズム

【図28】

```

1  for ( each row in the current frame ){
2    if ( this row is a MH row){
3      iterStatus = Inc_Mquant_Row;
4      for (iterCnt = 0; iterCnt <= IterMax; iterCnt++){
5        if ( iterCnt != 0){
6          if (iterStatus == Inc_Mquant_Row)
7            Increase the mquant values of all MBs in the row by delta;
8          else if (iterStatus == Inc_Mquant_MB)
9            Increase the mquant value of selected MB by 1;
10       }
11       Re-encode the whole row using the current mquant values for each MB;
12       if ( iterStatus == Final_Iteration )
13         Store the encoded bit stream for the whole row into the Row Buffer;
14       else
15         Only count the number of bits needed to re-encode row;
16       Byte Align the number of bits needed to re-encode row;
17       if (iterStatus == Final_Iteration){
18         Write video and mpeg information for MHs in row to output Fifo;
19         Exit the iteration loop;
20       }
21       else if ( BitsDiff < 0 ){
22         if (iterCnt == 0)
23           iterStatus = Final_Iteration;
24         else if (iterStatus == Inc_Mquant_Row){
25           Decrease the mquant values of all MBs by delta;
26           iterStatus = Inc_Mquant_MB;
27           Select the first MB in the row;
28         }
29         else if (iterStatus == Inc_Mquant_MB)
30           iterStatus = Final_Iteration;
31       }
32       else if ( BitsDiff == 0 )
33         iterStatus = Final_Iteration;
34       else if (iterStatus == Inc_Mquant_MB)
35         Select the next MB;
36     }
37     Perform bit stuffing to make OrigNumBits = RowBitCnt;
38     Replace original row with encoded bits from Row Buffer;
39   }
40 }

```

第2のMPEG再符号化方法のアルゴリズム

【図29】

```
1 Sum = PrevBitCnt ;
2
3 for ( i = 0; i < MB_width; i ++ ){
4     Sum = Sum + ( B1(k) - B1(k-1) );
5     if ( Sum <= RowBitCnt ){
6         j = i;
7         break;
8     }
9 }
```

k=(d+2)時のk番目の反復の場合に
figure4.1内にjを配置するアルゴリズム

【図30】

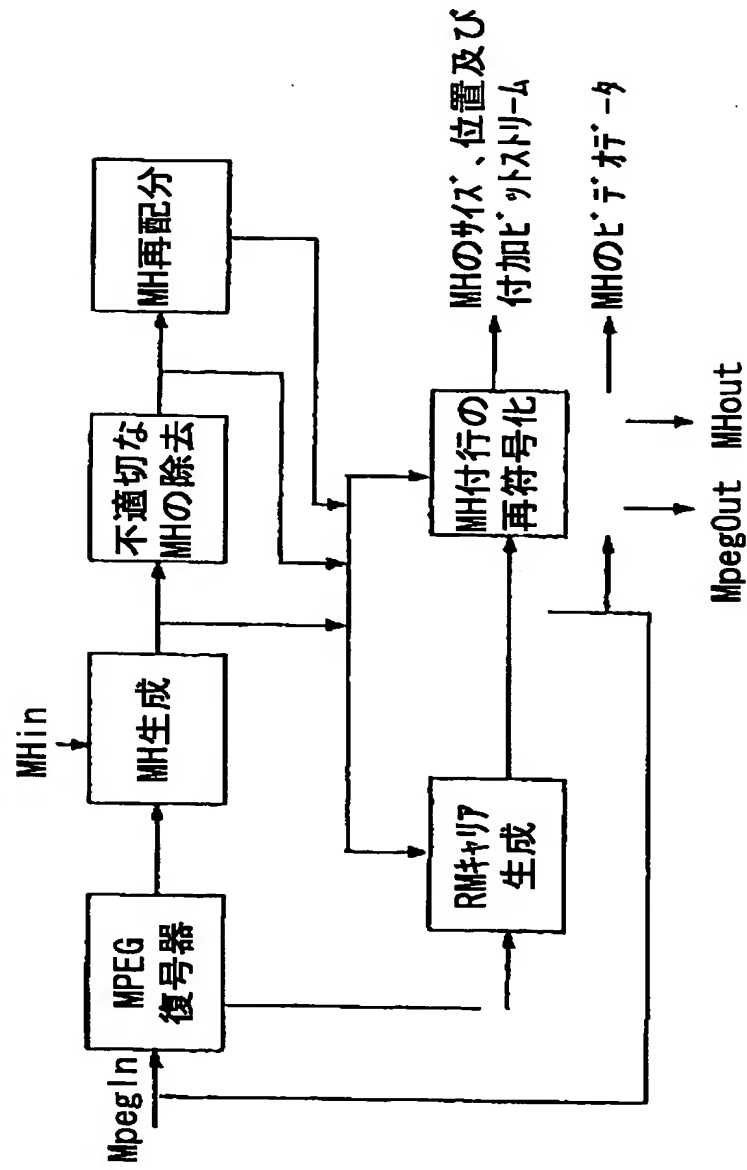
```

1  for ( each row of MBs in the current frame ){
2      if ( this row is a MH row){
3          iterStatus = Inc_Mquant_Row;
4          for (iterCnt = 0; iterCnt <= IterMax; iterCnt++){
5              if ( iterCnt != 0){
6                  if (iterStatus == Inc_Mquant_Row)
7                      Increase the mquant values of all MBs in the row by delta;
8                  else if (iterStatus == Inc_Mquant_MB)
9                      Increase the mquant value of selected MB by 1;
10             }
11             Re-encode the whole row using the current mquant values for each MB;
12             if ( iterStatus == Final_Iteration )
13                 Store the encoded bit stream for the whole row into the Row Buffer;
14             else
15                 Only count the number of bits needed to re-encode row;
16             Byte Align the number of bits needed to re-encode row;
17             if (iterStatus == Final_Iteration){
18                 Write video and mpeg information for MHs in the row to output Fifo;
19                 Exit the iteration loop;
20             }
21             else if ( BitsDiff < 0 ){
22                 if (iterCnt == 0)
23                     iterStatus = Final_Iteration;
24                 else if (iterStatus == Inc_Mquant_Row){
25                     iterStatus = Inc_Mquant_MB;
26                     Find the value of j as in figure 4.1;
27                     Decrease the mquant values of MBs by delta from j-th MB;
28                     Select j-th MB be the next MB to be updated;
29                 }
30                 else if (iterStatus == Inc_Mquant_MB)
31                     iterStatus = Final_Iteration;
32             }
33             else if ( BitsDiff == 0 )
34                 iterStatus = Final_Iteration;
35             else if (iterStatus == Inc_Mquant_MB)
36                 select next MB;
37             Save the value of the Row Buffer size to PrevBitCnt;
38         }
39         Perform bit stuffing to make OrigNumBits = RowBitCnt;
40         Replace original row with encoded bits from Row Buffer;
41     }
42 }

```

第3のMPEG再符号化方法のアルゴリズム

【図31】



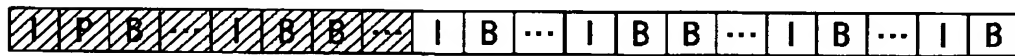
MH候補、RMキャリアの生成及び再符号化手順

【図34】

原ストリーム



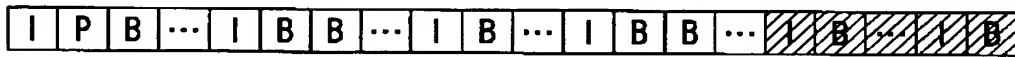
セグメント1:



セグメント2:



最終セグメント



【手続補正書】特許協力条約第34条補正の翻訳文提出書

【提出日】平成12年8月7日(2000. 8. 7)

【手続補正1】

【補正対象書類名】図面

【補正対象項目名】図1

【補正方法】変更

【補正内容】

【国際調査報告】

INTERNATIONAL SEARCH REPORT

International Application No.
PC1/US 99/11797

A. CLASSIFICATION OF SUBJECT MATTER IPC 6 H04N5/913		
According to International Patent Classification (IPC) or to both national classification and IPC		
B. FIELDS SEARCHED Minimum documentation searched (classification system followed by classification symbols) IPC 6 H04N		
Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched		
Electronic data base consulted during the international search (name of data base and, where practical, search terms used)		
C. DOCUMENTS CONSIDERED TO BE RELEVANT		
Category	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	WO 98 03014 A (PHILIPS ELECTRONICS N. V.) 22 January 1998 (1998-01-22) the whole document	1-3,5, 12,16, 17,19, 25,31, 36-38
A	WO 97 13248 A (PHILIPS ELECTRONICS N. V.) 10 April 1997 (1997-04-10) the whole document	1,3,4, 16,31, 36-39
-/-		
<input checked="" type="checkbox"/> Further documents are listed in the continuation of box C.		
<input checked="" type="checkbox"/> Patent family members are listed in annex.		
* Special categories of cited documents : "A" document defining the general state of the art which is not considered to be of particular relevance "E" earlier document but published on or after the international filing date "L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified) "O" document referring to an oral disclosure, use, exhibition or other means "P" document published prior to the international filing date but later than the priority date claimed "T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention "X" document of particular relevance: the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone "Y" document of particular relevance: the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art. "Z" document member of the same patent family		
Date of the actual completion of the international search 22 September 1999		Date of mailing of the international search report 28/09/1999
Name and mailing address of the ISA European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040. Tx. 31 851 epo nl. Fax (+31-70) 340-3016		Authorized officer Verleye, J

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

International Application No.
PCT/JS 99/11797

C. (Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT		
Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	<p>WO 96 06503 A (MACROVISION CORPORATION) 29 February 1996 (1996-02-29)</p> <p>page 3, line 10 -page 7, line 12 page 8, line 6 -page 9, line 8; figures 1-3</p>	<p>1-3, 16-18, 31, 33, 34, 36, 37</p>
A	<p>EP 0 107 567 A (THOMSON-CSF) 2 May 1984 (1984-05-02)</p> <p>page 3, line 14 -page 4, line 21; figure 3</p>	<p>1-3, 16, 18, 31, 36, 37</p>

I

Form PCT/ISA/210 (continuation of second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

Information on patent family members

International Application No.

PCT/US 99/11797

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 9803014 A	22-01-1998	EP 0940037 A US 5933798 A	08-09-1999 03-08-1999
WO 9713248 A	10-04-1997	CN 1166224 A EP 0795174 A JP 10510660 T	26-11-1997 17-09-1997 13-10-1998
WO 9606503 A	29-02-1996	US 5739864 A AU 698870 B AU 3370395 A BR 9508624 A CA 2195942 A CN 1160467 A EP 0777946 A EP 0853433 A JP 10504944 T NZ 292020 A US 5659613 A US 5668603 A	14-04-1998 12-11-1998 14-03-1996 18-11-1997 29-02-1996 24-09-1997 11-06-1997 15-07-1998 12-05-1998 28-10-1998 19-08-1997 16-09-1997
EP 107567 A	02-05-1984	FR 2534433 A	13-04-1984

フロントページの続き

(81)指定国 EP(AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OA(BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG), AP(GH, GM, KE, LS, MW, SD, SL, SZ, UG, ZW), EA(AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZA, ZW

- (72)発明者 コバド バグワディア
アメリカ合衆国 バージニア州 20165
スターリング、101、トリルムスクウェア
46229
- (72)発明者 ギョーム メルシエ
アメリカ合衆国 バージニア州 22102
マックリーン、リンカンウェイ 1504、ア
パートメント 437
- (72)発明者 シバ ラマドス
アメリカ合衆国 バージニア州 22102
マックリーン、リンカンウェイ 1504、ア
パートメント 437
- (72)発明者 マイケル ベルジュロン
アメリカ合衆国 バージニア州 23192
モントピーリア、ウィスパリングスプリン
グレーン 15150
- (72)発明者 ジャック エールハルト
アメリカ合衆国 バージニア州 23229-
4935 リッチモンド、センチュリードライ
ブ 7520

Fターム(参考) 5C053 FA13 GA07 GA11 GB06 GB15
GB37 JA24 KA21 KA24
5C059 KK43 MA22 MA23 MB27 RC04
RC09 RC31 RC35 RF02 RF05
RF18 RF27 RF30 SS13
5D044 AB05 AB07 BC06 CC04 DE03
DE40 GK08 GK17 HL08

**This Page is Inserted by IFW Indexing and Scanning
Operations and is not part of the Official Record**

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ BLACK BORDERS
- ☐ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☐ FADED TEXT OR DRAWING
- ☐ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☐ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☐ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☐ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: _____

IMAGES ARE BEST AVAILABLE COPY.

As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.